

AD-A076 146

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/6 17/7
ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQU--ETC(U)
AUG 79

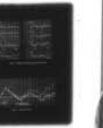
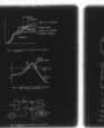
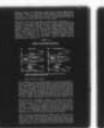
UNCLASSIFIED

AGARD-CP-272

NL

1 OF 4

ADA
076146



① LEVEL II

AGARD-CP-272

AGARD-CP-272

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AD A076146

AGARD CONFERENCE PROCEEDINGS No. 272

Advances in Guidance and Control Systems Using Digital Techniques

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

DDC
RECEIVED
NOV 5 1979
B

DDC FILE COPY

NORTH ATLANTIC TREATY ORGANIZATION



DISTRIBUTION AND AVAILABILITY
ON BACK COVER

NORTH ATLANTIC TREATY ORGANIZATION
ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT
(ORGANISATION DU TRAITE DE L'ATLANTIQUE NORD)

12 358

9
AGARD Conference Proceedings No. 272

6
ADVANCES IN GUIDANCE AND CONTROL SYSTEMS
USING DIGITAL TECHNIQUES

11 Aug 79

DDC
RECEIVED
NOV 5 1979
B

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

THE MISSION OF AGARD

The mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Exchanging of scientific and technical information;
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Providing scientific and technical advice and assistance to the North Atlantic Military Committee in the field of aerospace research and development;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.

The content of this publication has been reproduced directly from material supplied by AGARD or the authors.

Published August 1979

Copyright © AGARD 1979

All Rights Reserved

ISBN 92-835-0247-7



*Printed by Technical Editing and Reproduction Ltd
Harford House, 7-9 Charlotte St, London, W1P 1HD*

GUIDANCE AND CONTROL PANEL OFFICERS

Panel Chairman: Mr P.Kant, NL

Deputy Chairman: Mr G.C.Howell, UK

Panel Executive: Colonel J.C. de Chassey, FAF

PROGRAMME COMMITTEE FOR 28th GCP SYMPOSIUM

Chairman: Mr M.A.Ostgaard US

Members: Dr A.Benoit BE
Mr K.A.Peebles CA
Ing General M.J.Pelegri FR
Dr R.Ch.Onken GE
Dr Th.Spathopoulos GR
Mr P.Kant NL
Mr F.A.Østern NO
Mr J.L.Hollington UK
Mr W.F.Ball* US

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION _____	
BY _____	
DISTRIBUTION/AVAILABILITY CODES	
Dist.	AVAIL. and/or SPECIAL
A	

CONTENTS

PANEL AND PROGRAM OFFICERS

Page

iii

KEYNOTE ADDRESS

by E.J. Bobyn

vii

Reference

SESSION I – FUNCTIONAL DESIGN CONCEPTS, TRENDS, AND REQUIREMENTS

STATE OF THE ART FOR DIGITAL AVIONICS AND CONTROLS, 1978

by R.K. Smyth

1

A FLIGHT CONTROL SYSTEM USING THE DAIS ARCHITECTURE

by A.P. De Thomas and R.A. Hendrix

2

TRENDS IN DIGITAL DATA PROCESSING AND SYSTEM ARCHITECTURE

by A.A. Callaway

3

SESSION II – ADVANCES IN ANALYTICAL AND DESIGN TECHNIQUES

METHODOLOGIE DE CONCEPTION D'ARCHITECTURES MULTI-PROCESSEURS
POUR DES FONCTIONS D'AVIONIQUE

par C. Aleonard, A. Demoment, P. Romand, J. Gillon et J.F. Le Maitre

4

FORTRAN FOR AVIONICS

by A.J. Maher

5

AN OBSERVER SYSTEM FOR SENSOR FAILURE DETECTION AND ISOLATION
IN DIGITAL FLIGHT CONTROL SYSTEMS

by N. Stuckenberg

6

AUTOMATIC RECOVERY AFTER SENSOR FAILURE ON BOARD

by M. Labarrère, M. Pélegrin and M. Pircher

7

RECENT ADVANCES IN FIBRE OPTICS FOR HIGH INTEGRITY DIGITAL CONTROL
SYSTEMS

by R.P.G. Collinson

8

SESSION III – ADVANCES IN DIGITAL SYSTEM DESIGN AND ARCHITECTURE
TO ASSURE HIGH INTEGRITY

REDUNDANCY MANAGEMENT CONSIDERATIONS FOR A CONTROL-CONFIGURED
FIGHTER AIRCRAFT TRIPLEX DIGITAL FLY-BY-WIRE FLIGHT CONTROL SYSTEM

by J.H. Watson, W.J. Yousey and J.M. Railey

9

FAILURE DETECTION, ISOLATION AND INDICATION IN HIGHLY INTEGRATED
DIGITAL GUIDANCE AND CONTROL SYSTEM

by W.J. Kubbat

10

L'INTEGRITE DES LOGICIELS EMBARQUES: UNE SOLUTION

par G. Germain

11

A REDUNDANT INERTIAL NAVIGATION SYSTEM FOR IUS

by R.A. Baum, G.E.S. Morrison and R.C. Peters

12

DEFINITION OF THE HIERARCHICAL NETWORK FOR AGGRESSIVE
ENVIRONMENTS (RHEA)

by M. Buis, J.C. Laprie, J. Marco and D.R. Powell

13

PROTECTION OF A SIMPLEX DIGITAL COMPUTER FOR TERRAIN FOLLOWING*
by G.T.Shanks

14

SESSION IV - DATA PROCESSING AND COMPUTATION TECHNIQUES

DEVELOPMENT OF AIDING GPS/STRAP-DOWN INERTIAL NAVIGATION SYSTEM
by D.F.Liang, D.B.Reid, R.H.Johnson and B.G.Fletcher

15

DIGITAL ARRAY SIGNAL PROCESSING TECHNIQUES APPLIED TO GUIDANCE AND NAVIGATION
by S.Bloch

16

MICROCOMPUTER-BASED ON-LINE STATE ESTIMATION WITH APPLICATIONS TO SATELLITES
by N.K.Sinha, S.Y.Law and R.Mamen

17

METHODS FOR STRAP-DOWN ATTITUDE ESTIMATION AND NAVIGATION WITH ACCELEROMETERS
by R.P.Offereins and M.J.L.Tiernego

18

DIGITAL SIGNAL PROCESSING TECHNIQUES IN A MONOPULSE TRACKING RADAR
by U.Fazio, F.Ambrosioni and C. De Bonis

19

SESSION V - SOFTWARE DESIGN VALIDATION TECHNIQUES, INCLUDING SIMULATION

AN ASSESSMENT OF AND APPROACH TO THE VALIDATION OF DIGITAL FLIGHT CONTROL SYSTEMS
by D.B.Mulcare and W.G.Ness

20

VALIDATION OF GUIDANCE AND CONTROL SOFTWARE THROUGH REAL-TIME SIMULATION†
by R.G.Baldwin, J.P.Smith and W.C.Morton

21

LOGICIEL AVIONIQUE: EXPERIENCES PRATIQUES D'UNE METHODOLOGIE
par J.Perin

22

EXPERIENCE IN PRODUCING SOFTWARE FOR THE GROUND STATION OF A REMOTELY PILOTED HELICOPTER SYSTEM
by J.P.Webby, P.L.Wescott, M.I.Tucker and H.M.Smith

23

SIMULATION USE IN THE DEVELOPMENT AND VALIDATION OF HIMAT FLIGHT SOFTWARE
by A.Myers

24

SESSION VI - OPERATIONAL AND SYSTEM DEVELOPMENT EXPERIENCE

FEDERATED MICROCOMPUTER SYSTEMS FOR ON-BOARD MISSILE GUIDANCE AND CONTROL
by F.J.Langley, D.S.Siegel, W.Savage and R.E.Weelman

25

COPRA - UNE LIGNE NOUVELLE DE CALCULATEURS RECONFIGURABLES ULTRA-FIABLES DESTINEE AUX APPLICATIONS AEROSPATIALES EMBARQUEES (original French)
COPRA - A NEW LINE OF ULTRA-RELIABLE, RECONFIGURABLE COMPUTERS FOR AIRBORNE AEROSPACE APPLICATIONS
by C.Meraud and F.Browaeys

26

* Published in CP 272 Supplement (Classified).

† Not available at time of printing.

**ALIDADE – OPTIMISATION COUT-PERFORMANCE DE L'ALIGNEMENT D'UN SYSTEME
INERTIEL SUR PORTE-AVIONS***

par L.Camberlein, J.P.Paccard et M. de Cremiers

27

A HIGH ACCURACY FLIGHT PROFILE DETERMINING SYSTEM

by P.R.Vousden and P.J.Gollop

28

INTEGRATION OF FLIGHT AND FIRE CONTROL

by R.R.Huber

29

KEYNOTE ADDRESS

by

E.J. Bobyn
Canadian National Delegate to AGARD

Mr Chairman, Members of the Guidance and Control Panel, Ladies and Gentlemen:

Thank you Mr Chairman for that very kind introduction.

May I, first of all, as the Canadian National Delegate to AGARD, add my words of welcome to those that have already been expressed; Welcome to Canada; Welcome to Ottawa; I cannot say welcome to these elegant surroundings because this is not my building — except in my role as a Canadian taxpayer. It is the headquarters of the Canadian Department of External Affairs. You may have noticed on entering that the building is a rather rambling structure with a number of wings extending in all directions. It has been said that because of the very large number of senior officials in External Affairs, all of whom are entitled to corner offices, the building was designed to achieve the maximum number of outside corners.

My remarks this morning will be relatively brief because I know you want to get on with the business that brought you here — an examination of Digital Methods in Guidance and Control.

I would like to point out, however, that the general subject of Guidance and Control covers a much broader area than that which you will be discussing in this Symposium or, for that matter, the area under the purview of the Panel. For example, I can paraphrase my official terms of reference as Chief of Research and Development and say that I am responsible for the Guidance and Control of the R & D Program of the Department of National Defence. While this is not in the context of technology as applied to Guidance and Control, the field is still a very broad one even when restricted to the technological context. The Guidance and Control technology we deal with is applicable to environments other than aerospace — to naval, to land environments as well. Because of its fundamental nature we in Canada are very much concerned with issues in Guidance and Control.

We have Defence Research Establishments from coast to coast and each of them deal with technologies related to some aspects of Guidance and Control. Our principal activities take place in the DREs at Valcartier, Quebec and here in Ottawa. Within the broad definition of Guidance and Control, we have activities which include target acquisition and tracking, navigation technology, and new strapdown control techniques to cite examples. In this work we cannot hope to address all aspects of the topic; the area is clearly too broad for our resources. We do undertake effort in those areas which we feel are especially important to Canada and in which we feel we have both unique needs and expertise. Our contribution of the results of our research and development to the international defence community is something in which we feel a justified pride and towards which we are fully committed. In the sessions to follow, you will hear several Canadian papers; they are representative of our national efforts but of course they do not tell the whole story. Our efforts, as are those of the other countries, are both broad and constantly changing. From day to day the problems change; new hardware, new materials, new design concepts, new mathematical solutions come from laboratories of the Alliance almost daily. Solutions derived for a problem not normally associated with Guidance and Control often facilitate new methods applicable directly to it. For example, the evolution of microprocessors has made possible the strapdown inertial systems now becoming attractive for use in guidance and navigation systems; the development of extremely stable cesium clocks resulted in the precise satellite navigation system "NAVSTAR" which is now being developed, and in which NATO is making a contribution. Thus we live in a world filled with change and challenge. There is a certain contradiction inherent in this aspect of our lives because we have grown so used to change, so accustomed to innovation, that they now seem commonplace. It is very important to retain one's sense of perspective in the technological progress we witness, for this can stimulate us to move further in bringing the growth of Guidance and Control to full maturity.

To retain this sense, to keep a sense of our progress, two attributes are necessary. One is an ability to step back and see our present state in terms of our origins (how far we have come) to observe the evolution of Guidance and Control. The other attribute is an ability to see clearly the component elements which constitute this broad field, the ability to analyze (and thus appreciate) those factors. I would like to illustrate these points in a bit more detail.

NATO is now in the process of adopting the Airborne Warning and Command System — the AWACS. With such a system, a military commander will have the ability to collect tactical information in real time. He will be able to observe

the progress of constantly changing tactical air forces, movements, and engagements; he will be able to deploy his forces as required based on precise knowledge of the existing situation. The information he will be able to draw upon for his decision is far more extensive than he will be able to use — it must be filtered, selectively chosen, for the information volume exceeds the ability of the human mind to operate upon it. To examine the implications of this, we can compare this commander's situation with what his counterparts faced in earlier times. Commanders of 100 or more years ago had to operate with only partial information which arrived days, weeks, and sometimes months late. The forces which they commanded had to operate with an equivalent lack of information, they moved through dust, fog and clouds, out of touch with the forces on either side. uncertain of their position, unsure of where their enemy lay. This situation was pertinent to military operations well into this century. That successful operations could be accomplished in such a difficult environment is a source of wonder. But during and since the Second World War we have seen a rapid surge in creative effort which has brought us to where we stand today, to the AWACS, to the new inertial guidance technology, to the microprocessors, to the low light cameras which allow us to see what our eyes cannot, and to the host of other advances which have emerged from aerospace research and development. For further examples, in the 1930's Werner Von Braun was studying simple inertial guidance and control systems, trying to recover rockets to determine what fault had caused them to lose control; in the 1940's, an electronic digital analyzer was constructed at the University of Pennsylvania — it was slow, and it filled a large room. There are many such examples but these illustrate my point — that our present environment is better seen and appreciated by a sense of the past.

I mentioned that there were two attributes which enhance our ability to appreciate our current position and which stimulate further effort. The second is that of analysis — the subdivision of a broad technology area such as guidance and control into its constituent elements. There are many technologies that make Guidance and Control effective. They include sensors — video, electromagnetic, inertial, pressure; they include displays — head up, color, and storage; and they include digital techniques, the subject of this symposium. In all the basic technologies, digital methods are providing great advances in capability. I spoke of operations based on partial information. With the electronic age came analogue techniques, an extension of the way in which humans have always operated. The development of digital techniques has been both responsible for and driven by modern technology requirements. With digital methods one can carry out many operations very quickly. The ability to do this has led to the need to carry out still more operations and this in turn has led to the development of faster electronics, more efficient software (and hardware), and smaller, less expensive systems. Digital techniques are found everywhere — data collection, data transmission, and data processing have reached the position today that the human who uses the information must be very selective in choosing what data he wants to examine. He is in danger of having too much available.

That the problems of guidance and control have not all been solved by the application of digital techniques is evidenced by this symposium. Those of you who are intimately involved in the field are well aware of the problem areas. Within the next few days you will be listening to others discuss their success in overcoming problems, or approaches to them. You are here both to learn and to exchange information. It is through such an exchange and through application of your skills to the problem areas that we will advance yet further in Guidance and Control technology, and there are advances to be made. At the most recent G&C symposium there were discussions concerning guidance and control of helicopters — at night and in poor visibility. There have been tremendous advances in the sensors, processing, and display of information — advances which in fact allow one to "see" terrain not normally visible, and thus pilot a helicopter over it. The techniques which permit this are largely digital in nature, and while they work well, there is a reluctance on the part of the operators to trust the information they receive. They do not like to pilot their aircraft over terrain they cannot actually see. Thus we still have a psychological wall to overcome, a wall which divides our analog daily lives from the efficient, sensitive, fast world of digital techniques. This is an important factor to consider as we attempt to accomplish our goals digitally. Whenever the human operator enters, what is needed is a close approximation to what he perceives as the real world. Without overcoming this inherent mistrust in a "reconstituted" world, our advances will have a major element missing. Simulators, training, and experience will help to overcome this problem.

There are other problems which challenge you. Size, space, and cost are all parameters to be reduced as much as possible. Inherent in many digital techniques — those of command and video data links for example, is the use of relatively low signal levels. For smart weapons, remotely piloted vehicles, satcom terminals, etc, these data links can be vulnerable to jamming. The vulnerability can be lessened through techniques such as bandwidth compression but with a loss of information. This loss can have an impact on the operators reliance on the information — and we again return to the situation I mentioned before — our ability to rely on information which has been transformed by technological requirements. The emergence of cruise missiles presents an enormous challenge in Guidance and Control. The requirements which will enable such a missile to fly for many hundreds of miles, close to the earth, while avoiding its hazards, are problems which would have had no practical solution just a few years ago. Today we can successfully tackle them. Further off in time, but not visible, are other concepts — manned aircraft will be controlled almost totally by electronics; sensors and computers on board will determine aerodynamic information, and wing warping of fighters in flight will be used to obtain greater performance than is now obtainable. New materials, geometries, and designs will place new demands on the aircraft control techniques, demands which we are only now becoming aware of. Weapons-pointing in high dynamic aircraft is another example; Servo mechanization of weapons would be synthesized in the flight control system. The high dynamic fighter may have specific control laws preprogrammed into it, laws which are applicable to specific mission segments. The pilot might then use a CRT to select, from an adaptive "menu" of available laws, the optimum combat mode he desires. Ladies and gentlemen there are many future possibilities. You are aware of them I know.

Before closing I would like to reemphasize my main point. The field of Guidance and Control is one of enormous challenge. We face many difficulties in addressing these challenges and the solutions are very important indeed to the NATO alliance. When one is very closely involved in a field it is only too easy to see only the problems — to become so close to the situation that the broader view is difficult to achieve. Successful solution of the problems we face is far easier if we have a sense of enthusiasm and accomplishment. I have illustrated the distance we have come together, the rapidity of our progress, and some of the challenges which lie ahead. Within the next four days we will meet some of the challenges. We must listen to what is said. We must consider the applicability of the papers to our own problems. And we must consider and propose ways in which these efforts can be applied within the Alliance for the common goals we work towards. I wish you every success in this important and exciting effort, and a stimulating four days of discussion during your symposium.

Thank you.

STATE OF THE ART FOR¹
DIGITAL AVIONICS & CONTROLS, 1978

DR. RICHARD K. SMYTH
MILCO INTERNATIONAL, INC.
Huntington Beach, California

SUMMARY

This paper provides a brief summary of a comprehensive State of the Art Survey conducted for NASA headquarters in 1978 in support of a 10 year NASA Research and Technology Plan for Avionics and Controls directed by Dr. Hermann A. Rediess of NASA Headquarters. The Survey was made by contributions from some 72 individuals, whose names are listed in reference (1), from US Industry, Universities, and Government agencies. The Survey includes 5 broadly applicable technology areas: Flight Path Management, Aircraft Control Systems, Crew Station & Human Factors, Integration & Interfacing Technology, and Fundamental Technology. In addition the survey included military technologies which have a technology transfer potential to the five broadly applicable technology areas.

The State of the Art Survey also predicts technology trends.

1.0 INTRODUCTION

This State of the Art Survey is in support of the NASA OAST planning committee for Aeronautics, Avionics, and Controls headed by Dr. Herman Rediess. NASA OAST sponsored a Workshop for Avionics and Controls held near NASA Langley Research Center 27-29 June 1978. This workshop included invitees from Industry, the Universities, various NASA Centers, FAA and other Government agencies. The workshop began with a plenary session during which the ground rules were outlined. A three hour oral summary of this SOAS Report was presented by the Editor. The workshop divided into committees which considered the issues of the six major sections of this SOAS Report including:

- FLIGHT PATH MANAGEMENT TECHNOLOGIES
- AIRCRAFT CONTROL SYSTEM TECHNOLOGY
- CREW STATION TECHNOLOGY
- INTEGRATION & INTERFACING TECHNOLOGIES
- MILITARY AVIONICS TECHNOLOGY
- FUNDAMENTAL TECHNOLOGIES

The principal objective of this State of the Art Survey (SOAS) Report is to identify the current State of the Art in each of these technology areas and to identify emerging trends which are important to the future development of these technologies. This SOAS Report will help the NASA OAST planning and study committee fulfill its charge to define just what NASA's role should be in these technologies during the next two decades.

The timing for this State of the Art Survey is particularly appropriate coming at the time when a virtual revolution in microprocessor technology is taking place which promises to have a profound impact upon the way avionics and control systems are implemented. There are a number of System technologies which are emerging which promise to contribute an even greater impact upon the way we structure the avionic and control system architecture of the next generation systems. These trends include:

NAVSTAR/GPS which promises to provide a common, accurate, worldwide navigation system, which may replace many of the current navigation systems. There are many painful trade-offs and transition decisions which must be made before such a change can occur. The complete system validation is still some seven years away.

AIR TRAFFIC CONTROL UPDATE which includes a Discrete Address Beacon System (DABS) with a data link capability, Beacon Collision Avoidance System (BCAS), Automatic Traffic Advisory and Resolution System (ATARS), and Microwave Landing System (MLS). These ATC improvements and others being considered will have a rippling effect on all aircraft avionics.

DIGITAL FLY BY WIRE SYSTEMS (DFBWS) are imposing new looks in fault tolerant hardware and software configurations which promise to reduce aircraft weight and cost without compromising flight safety or reliability.

¹ This work was supported by the National Aeronautics & Space Administration headquarters under Contract NASW-2691. MILCO performed the effort under Subcontract 7037 to ORI, Inc, Silver Springs, Md.

THE ADVENT OF FULL AUTHORITY DIGITAL PROPULSION CONTROL SYSTEMS and their integration with digital automatic flight control suggests new economies and performance benefits to both propulsion and flight control systems.

THE INTEGRATION OF DIGITAL COMPUTERS with flat surface displays and integrated data control centers is producing a revolution in cockpit avionics. The crew station systems technology is providing control of more modes and functions in less panel space through the use of interactive display and control techniques. The introduction of human factors into crew station design at an early stage is simplifying the operational procedures from the crew's point of view.

THE TREND TOWARD MODULAR AVIONICS architecture with distributed microcomputers is accelerating and introducing new levels of flexibility to aircraft avionics. The digital multiplex bus concepts are making needed information accessible to all modular functional elements which makes for ease of system change and functional expansion.

THE INCREASING CAPACITY AND DECREASING COST of airborne information processing is having a dramatic impact upon the sophistication in avionics functions and capabilities even for low cost General Aviation aircraft which comprise the vast majority of all aircraft flying.

THE FEASIBILITY OF FIBER OPTICS for aircraft avionics interconnect and busses appears probable during the next decade. The replacement of copper wire bundles with fiber optics bundles will have a significant impact upon reducing avionics wiring weight and elimination of EMI and indirect lighting effects.

MILITARY AVIONICS TECHNOLOGY continues to provide system concepts and technology transfer elements to mission areas which fall within the responsibility of NASA OAST. For example, the military is on the leading edge of high speed digital processing applications used for such missions as hostile signal intercept, radar signal processing, and imaging guidance sensors. The high speed processing elements are already in the nanosec region and are progressing toward the picosecond region. Also the Military Avionics Technology is contributing to the State of the Art of high density electronics packaging which must survive a severe temperature, shock, and acceleration environment, such as the US Army Cannon Launched Guided Projectile (CLGP). These mass quantity programs such as CLGP must also result in designs which have a low unit cost.

MODERN CONTROL THEORY and Systems Analysis Techniques developed during the last quarter century are providing synergistic benefits to system design brought about by the low cost digital processing available through microprocessor technology, low cost memory, and new high speed mass memory technologies such as bubble memories and EBAM. Techniques which were at one time belived by practitioners of system design to be too complex (and not understandable) are now commonplace in current and emerging avionics and control systems. These modern control theory techniques include:

KALMAN FILTERS AND STATE ESTIMATION TECHNIQUES which provide filtering of data and accurate prediction of state variables made possible by analytical models of the vehicle and its controllers. These techniques have been applied to navigation, guidance and control systems and are becoming commonplace even on General Aviation avionics.

OPTIMAL CONTROLLERS which use cost functions to synthesize new system configurations to solve a wide array of complex control problems.

DIRECT DIGITAL SYNTHESIS is receiving renewed emphasis as real time digital control systems proliferate in the next generation avionics and controls designs. The challenge of multiloop, multi-sampling rate, multi variable, time varying, distributed systems provides for great potential payoff of new methods.

THE USE OF LARGE SCALE DIGITAL COMPUTERS to simulate complex real time digital-hybrid systems is, of course, commonplace. A trend toward using flexible, programmable breadboards of the actual system is emerging which relieves some problems of fidelity of simulation in distributed digital systems.

THE PROPORTION OF SOFTWARE costs to hardware costs is growing with each generation of new systems using low cost digital microprocessor/memory technology. As a result, widespread concern over the the growing costs of software has caused a large groundswell in the avionics industry aimed at curbing the escalating costs of software. This effort to reduce software costs has resulted in a collection of techniques referred to as 'Modern Software Programming Practices' (MSPP). MSPP has

produced the following trends in Avionics Software Design:

- The use of Higher Order Languages for Avionics to replace the previous technique of using assembly language programming in the real time systems of avionics.
- Top Down Software Development which defines the software as a hierarchy of levels.
- Structured Programming which defines independent software modules with one entry and one exit.
- Modular Software associated with distributed microcomputer architectures.

THE DISPLAY RESEARCH AND DEVELOPMENT taking place to find a viable replacement for CRT displays in the cockpit is accelerating with a number of possible candidates on the horizon including Liquid Crystal Displays (LCD's), Light Emitting Diodes (LED), Gas Plasma Displays as well as exotic new technologies such as electrochromic displays, electrophoretic displays, and electroluminescent displays. To date a clear winner in the candidate to replace CRT's has not emerged.

This Avionics and Controls State of the Art Survey (SOAS) Report addresses the six technologies listed on the first page of the Introduction. The Survey was conducted in a short time. However, the large cross section of Industry, Government, and Universities represented by the individual contributors provides a unique snapshot of avionics and controls technology in 1978. Equally important, however, since these more than 60 contributors are also working on the next generation avionics and control systems, their write-ups and viewpoints provide a preview of the future in these technologies.

The final version of this report was impacted by the deliberations and conclusions of the more than 100 participants at the NASA sponsored Avionics and Controls Workshop held at Hampton, Virginia, near NASA Langley on 27-29 June 1978. The working sessions of the workshop were organized into parallel groups which considered five of the six major technology areas (all except Military Avionics Technology) covered by this SOAS Report.

2.0 FLIGHT PATH MANAGEMENT TECHNOLOGIES

This technology area covers the navigation, guidance, communication, and air traffic control aspects of aircraft flight path management. This section also covers the means for avoiding hazardous weather conditions in flight. The civil aircraft considered by this technology include the broad spectrum from the simplest general aviation aircraft to the most modern wide body jet transport. The current state of the art of flight path technology is based upon a number of national navigation and air traffic control systems which will be changing during the next decade.

2.1 NAVIGATION AND POSITION FIXING

The backbone of civil aviation navigation is the visual omni range (VOR) and distance measuring equipment (DME) network which is now implemented world wide for over land navigation. The airlines widely utilize inertial navigation systems (INS) for transoceanic flights to supplement VOR/DME in areas of sparse coverage. Another self-contained navigation system used on transoceanic flights is doppler radar which suffers from larger errors than INS. Both INS and doppler require position updates to bound the time increasing errors. For example, INS typical error rate is approximately 1 NM/hr for most of the widely employed civil units.

The primary navigation systems are being supplemented by other radio navigation aids and, to a limited extent, by non directional radio beacons whose primary use is for ILS acquisition. Omega is perhaps the most widely used of the other radio navigation aids, since it provides world wide coverage particularly when supplemented by the Navy's VLF communication stations. Seven of the eight Omega stations are operational with Australia's scheduled for operation in the near future.

Loran C is another long range radio navigation system which has found limited use in civil aviation primarily because of its limited geographical coverage to date. The principal use has been in the coastal waters and in the northern Atlantic and the Middle East.

The new satellite navigation system being developed by DoD, NAVSTAR Global Positioning System (GPS), promises to replace the existing

navigation system in time and provide accurate, world navigation. The estimates for when NAVSTAR/GPS will replace VOR/DME range from 1985 to 1995. The GAO (ref 1) has recommended an early replacement of other navigation systems by NAVSTAR/GPS to stop the proliferation of overlapping navigation systems. However, other government agencies have challenged the feasibility of an early replacement primarily because of the early state of validation of GPS and the enormous effort required to change the navigation infrastructure.

Without entering the debate on the date of switch over, it does appear that before the end of the century it is highly probable that NAVSTAR/GPS will replace VOR/DME and the other radio navigation systems. The technology in receivers and digital processing are progressing at a rapid rate which should prove the most optimistic cost estimates for GPS receivers correct in the long run.

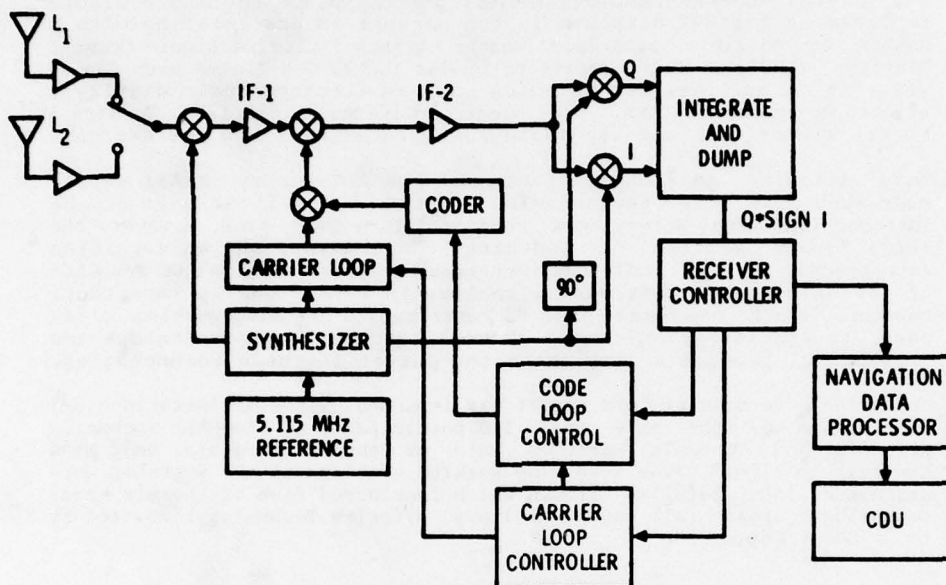


FIGURE 2.1-1 GPS USER EQUIPMENT FUNCTIONAL DIAGRAM

A significant trend toward strapdown inertial systems has developed which may yield simpler, lower cost systems. New technology gyros employing ring laser techniques are developing. One strapdown system uses electrostatically suspended gyros. The most common gyro employed in the current generation of INS is the tuned rotor gyro.

2.2 GUIDANCE

Comments on guidance section will be limited to outer loop control law aspects. Other aspects of guidance such as flight path definition, processor, and display technology, and signals in space are covered in sections 4.3, 5.2.1, 4.1 and 7.5 and 2.1, respectively. As far as critical technology is concerned it is felt that flight path definition and control/display technology (amenable to ease of pilot/ATC controller and pilot/computer interaction) are the critical guidance technology areas.

RNAV and VNAV should be discussed in enroute, terminal and final approach areas, the latter being subdivided into narrow (ILS or low cost MLS) or wide angle (MLS) final approach guidance.

Lateral guidance techniques have approached a level of maturity where only fine tuning is required. Enroute and terminal area accuracies and ride qualities can be achieved from practically all recognized position fixing systems in single VOR/DME or multisensor configurations. (ref 2) Final approach guidance, even up to Category III, is not so much a problem of control law development as it is one of revisionary management leading all the way down to pilot awareness and takeover in the event of system failure. The major problems in the latter event are pilot factors and display technology. Acceptable means of transitioning from enroute/terminal nav aids to final approach aids is also more a problem of providing acceptable means for the pilot to monitor progress than it is one of the control system providing performance. (ref 3) This is true whether the transition is to a wide angle or a narrow beam final approach guidance system. (ref 4) The procedures may vary somewhat but the control laws probably will not since the signals in space have approximately the same accuracy and

stability characteristics. Aircraft 4D approach control has also been shown to be feasible for different complexities of ground and airborne equipment. (ref 5, 6) Incompatibility of procedures for aircraft equipped with sophisticated guidance systems and for those minimally equipped appears to be the major stumbling block inhibiting implementation of these techniques.

Figure 2.2.1-2 shows a VOR/DME based RNAV system which utilizes waypoints defined by latitude and longitude. The computer automatically tunes to the proper VOR/DME stations based upon aircraft location. The unit stores the latitude and longitude of every VOR station in the world which it can access for RNAV computation.

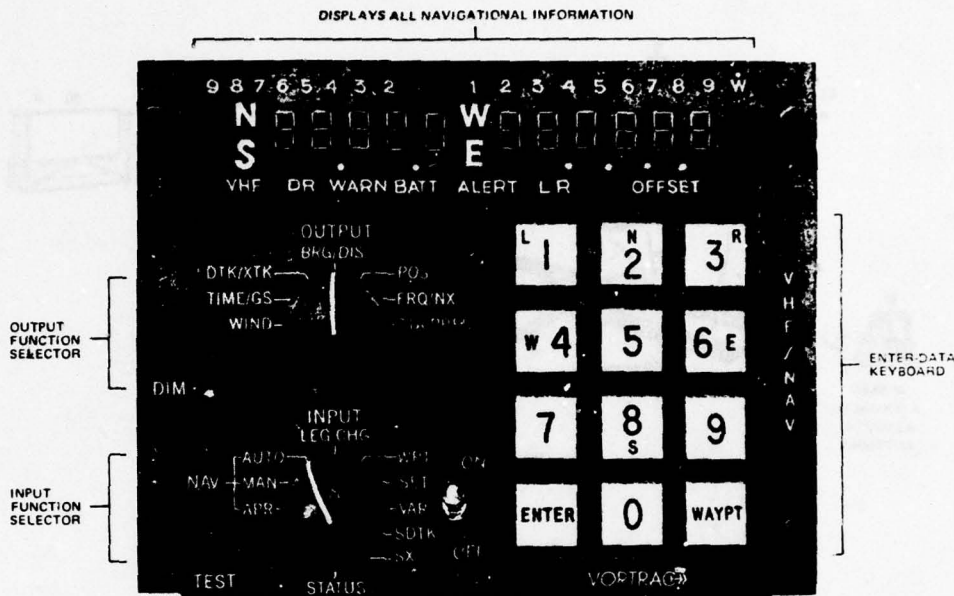


Figure 2.2.1-2 Typical VOR/DME R-NAV system utilizing LAT/LNG waypoint definition.

MICROWAVE LANDING SYSTEM

Purpose

The Microwave Landing System (MLS) is an electronic aid to aircraft navigation during the approach and landing at an airport runway. Its purpose is to provide a common civil/military approach and landing system with improved performance and more flexibility for implementation than existing systems.

In 1971, a joint development program was initiated by the Department of Transportation (DOT), the Department of Defense (DOD) and the National Aeronautics and Space Administration (NASA) (ref. 9). This joint development was undertaken under FAA management to provide a National standard for a family of compatible landing system configurations which would be adaptable to the needs of a wide range of civil and military users. The system has also been designed to meet the international need for a new standard Non-Visual Approach and Landing Guidance System as a replacement for the existing ILS (ref. 10).

The system has been designed so compatible system elements can be combined to meet the needs of a specific airport facility. The three major configurations were identified for prototype development. These are: (a) basic, (b) expanded, and (c) small community. Illustrations of these configurations are shown in Figure 2.2.5-1. (Ref. 11)

2.3 AIR TRAFFIC CONTROL

Major Technical Developments (near-term)

Aircraft Separation Assurance

The FAA is engaged in three major development activities as backup safeguards against midair collisions. These are known as Conflict Alert, Beacon Collision Avoidance System (BCAS) and Automatic Traffic Advisory and Resolution Service (ATARS).

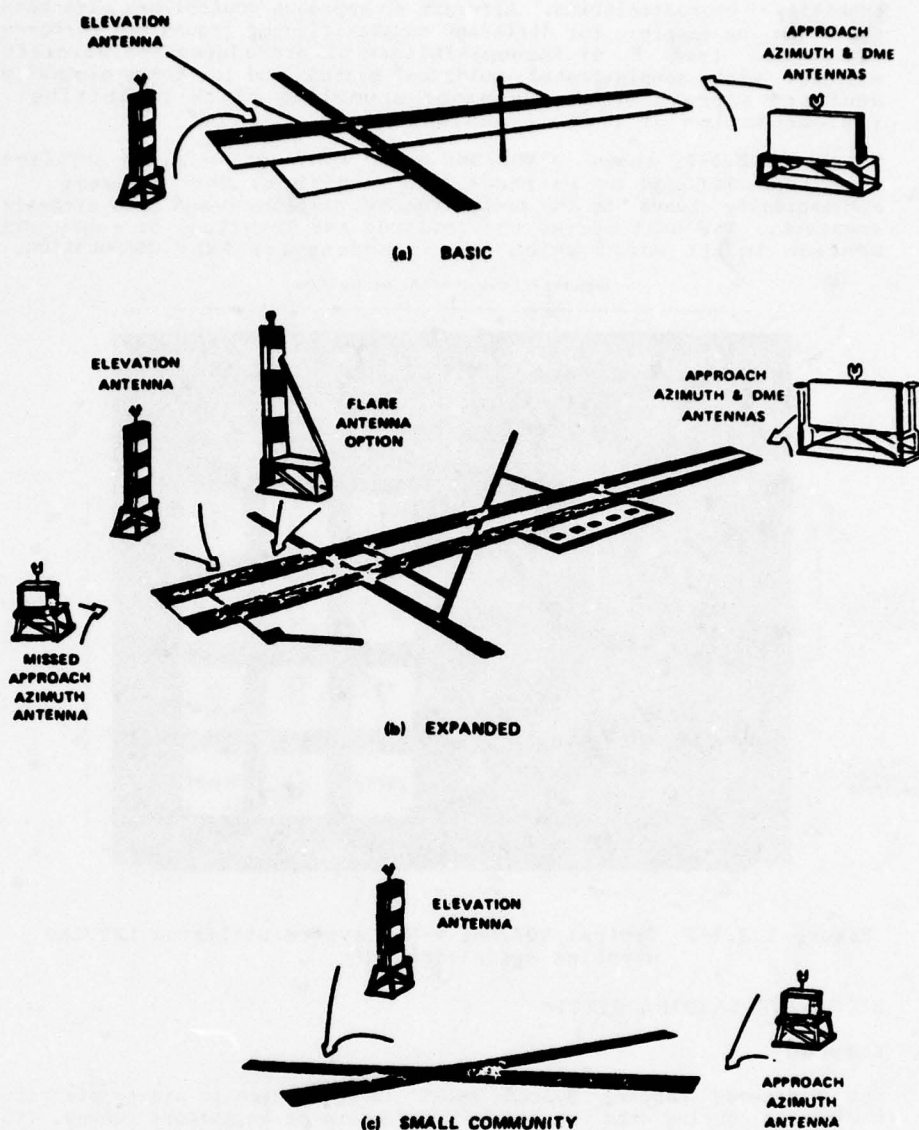


FIGURE 2.2.5-1 EXAMPLES OF TYPICAL GROUND CONFIGURATIONS

Conflict Alert.

The current software development program for the terminal and enroute computerized control systems includes new features to detect possible air traffic conflicts. These improvements in the processing of track information on aircraft with altitude reporting transponders alert the controller to possible traffic conflicts.

The Beacon Collision Avoidance System (BCAS)

This system is being developed to provide an operationally sound all-weather airborne collision avoidance service by the FAA issuing a U.S. National Standard for BCAS avionics. The system is based upon use of airborne beacon transponders and is independent from the primary ATC system. Any aircraft operator desiring the BCAS service can install appropriate BCAS avionics when fully developed and receive warning of impending collision or near-miss threat from nearby aircraft carrying an altitude encoding transponder as the cooperating airborne element.

Automatic Traffic Advisory and Resolution Service (ATARS)

This is a conflict alert and resolution service for use by pilots of ATARS-equipped aircraft. This capability is being developed concurrently with DABS and will use the DABS data link for transmission of ground-derived separation assurance information. If potential traffic conflicts are not resolved in a predetermined time frame by the basic ATC system or independent pilot actions, a ground

based computer processing DABS data will be capable of issuing separation maneuvering commands directly to the aircraft involved.

Discrete Address Beacon System (DABS)

This is a ground based surveillance system with an integral data link being developed by the FAA to significantly improve the quality of ATC surveillance data. DABS is being designed as a compatible replacement for the present ATCRBS facilities. The new system will be able to discretely address aircraft equipped with DABS transponders to eliminate synchronous garble of beacon responses in high density areas. An integral two-way data link is being designed into the system for control messages and other air-ground communications.

Airport Surface Traffic Control

The Airport Surface Traffic Control (ASTC) program is two-phased to improve the surveillance and ground control of surface traffic at major airports. The first phase of the program is to replace existing tower radar equipment with new Airport Surface Detection Equipment (ASDE). The ASDE-3 now under development will increase system reliability and provide enhanced target detection and display capabilities to assist tower control of surface traffic in periods of reduced visibility.

Wake Vortex Programs

A potentially hazardous condition can exist for aircraft departures and approaches to landing when encountering trailing vortices from other aircraft. Under certain atmospheric conditions and light winds, strong vortices generated by large jet aircraft dissipate rather slowly and can make aircraft encountering the vortices difficult to control. The potential existence of this hazard has required changes in aircraft separation standards and resulting reductions in airport capacity at jet airports.

Flight Service Station Automation

to provide service facilities for airmen to obtain flight planning information and file flight plans. The FAA is engaged in a major program to automate many of the FSS functions and provide more efficient means for direct access by users to improve services and reduce system costs. This program involves the development of a new system configuration for automation, consolidation and collocation of FSS facilities.

Increased ATC Automation

The FAA is engaged in hardware and software developments to provide new automation features in the enroute and terminal control systems. The bases for these programs are the NAS Stage A automation system and the ARTS III system. The NAS Stage A system is implemented at the 20 enroute control centers in the continental US (ref 16). It processes flight plan information and aircraft track information on all controlled aircraft. The ARTS III system is installed at 63 medium and high density terminal control centers serving major airports. Several new features are being developed to upgrade the capabilities of these automated control systems.

2.4 COMMUNICATIONS

The current civil communications is based upon VHF voice communications using 720 channels in the spectrum between 118.0 and 136.0 MHz. In addition voice communications are available on HF in the frequency range from 2.8 to 26 MHz. At the present time data link is unavailable to civil aviation except on a commercial VHF channel operated by ARINC for airline operational data transfers.

The DABS system will provide a data link capability between aircraft and ATC which will have an effective baud rate of about 2400 for extended uplink and down link messages. The advent of DABS data link should reduce the amount of voice 'clutter' during ATC operations and provide for unambiguous display of ATC clearances and acknowledgements on cockpit and ground CRT (or flat surface) displays.

2.5 WEATHER AVOIDANCE

Weather represents the major uncontrollable variable affecting the operation of the air traffic control (ATC) system. It is the largest single causal factor in the delays encountered in the National Airspace System (NAS), and is a major contributor to aviation accidents. Even

with planned improvements in ATC system capacity, costs attributable to operating delays may reach \$1 billion in the next several years. The safety of flight as well as operating efficiency is highly dependent on the terminal area ATC system, both in the air and on the ground. The ATC system improvement and automation programs will include weather data for use by traffic controllers and for transmittal to pilots.

To minimize or avoid the impact of weather factors on aircraft operations requires the capability to predict or forecast weather conditions for both short-term and long-term periods. It also requires the availability of sensors and supporting systems to acquire, process, transmit and display weather data relevant to aviation needs, especially as related to severe weather phenomena. To achieve these capabilities, the Weather Avoidance programs have been divided into three general categories: wind shear, automation of weather observation, and weather data processing and distribution. These are funded under the Research, Engineering and Development appropriation request, along with related air traffic control, navigation, and aviation medicine engineering and development programs. Each of the programs is structured into a number of subprogram areas and is conducted by FAA personnel, other Government agencies by means of interagency agreements, or through contracts with qualified organizations.

3.0 AUTOMATIC CONTROL SYSTEM TECHNOLOGY

Automatic control system technology is undergoing a significant rate of technology turnover which at first seems surprising for such a mature technology. The technology turnover is caused by several factors which include:

- A strong trend toward digital mechanizations of control systems brought about by the rapidly decreasing cost and increasing computational power made possible by micro-processor technology and other very large scale integration (VLSI) circuits.
- A trend toward increasing the functions of flight control systems to include such active control modes as flutter control, gust alleviation, and ride quality enhancement.
- The trend to fly by wire systems to reduce aircraft weight is requiring the development of fault tolerant computer hardware and software to retain the same levels of safety and reliability as for the mechanical systems being replaced.
- Digital electronic controls are now being applied to propulsion controls in order to achieve additional performance and economy benefits not possible with the mature hydromechanical technology.
- The integration of propulsion and flight control is a natural consequence of the digital implementations of these control loops.

3.1 AUTOMATIC FLIGHT CONTROL SYSTEM

Primary flight control systems for aircraft have evolved from simple mechanical linkages between the pilot's controllers and control surfaces to electronic (Fly-by-Wire) flight control systems with fully powered control surfaces. Aircraft flight envelope expansion and increased response requirements have given impetus to this evolution. Improvements in reliability and miniaturization of electronics, coupled with technical advances in hydraulic actuation, have made it possible to incorporate flight control systems which can provide significant aircraft performance improvements. The evolution of flight control systems is depicted in Figure 3.1-1. Each of the flight control system types illustrated in Figure 3.1-1 are presently in use or being designed for use in future aircraft.

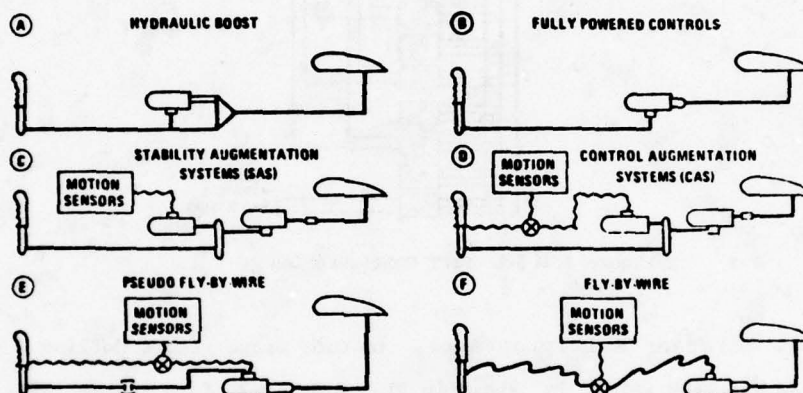
Fly-by-Wire (FBW) controls were initially developed and flight tested on the MCAIR F-4 Survivable Flight Control System (SFCS) program funded by AFFDL and subsequently implemented in production by General Dynamics on the F-16 aircraft. Sperry supplied the redundant analog system. FBW technology makes possible the implementation of the active control capabilities in a more cost effective manner. Digital mechanization of FBW avionics provides the flexibility and computational capacity to implement these capabilities in a more cost effective manner. Digital avionics were flown on the LTV A-7 aircraft in an AFFDL funded program to Honeywell. The mechanical controls in the A-7 aircraft were retained. NASA is currently developing a triplex digital FBW with an analog FBW backup for an LTV F-8 aircraft. In

addition Boeing is developing a triplex digital control system with a mechanical backup for the AMST using Marconi Elliott flight control electronics. The first production digital system is being developed for the Navy F-18 fighter/attacker aircraft built by the MCAIR and the Northrop team with flight control avionics supplied by G.E. The F-18 incorporates a minimal mechanical backup system.

Customized flight control modes can be designed and implemented effectively in a digital FBW flight control system to provide enhanced tactical effectiveness. Modes that optimize velocity vector control for delivery of ballistic free-fall air-to-ground weapons and optimize attitude control for gun aiming in aerial combat missions are primary examples. Coupling of the digital FBW system to the fire control system can provide significant improvements in weapon delivery accuracy and aircraft survivability for both aerial combat and air-to-ground attacks. Utilization of an aircraft configuration with direct force control in conjunction with the digital FBW technology enables the improvement of maneuvering performance through relaxed static stability, flat turning, vertical path control, fuselage aiming and vertical/lateral translation.

FIGURE 3.1-1

HOW FLY-BY-WIRE EVOLVED



(FROM J. P. SUTHERLAND AND R. C. HENDRICK, "ELECTRONIC FLIGHT CONTROL IS GETTING SET TO TAKE OFF", ELECTRONICS, NOVEMBER 9, 1970, PP 87-92)

FAULT TOLERANT SYSTEMS

The development of fault tolerant systems is one of the most important efforts in adapting advanced technology to aeronautical applications. Recent developments, particularly in micro-electronic technology, give the potential for much more capable and cost effective avionics systems than the ones currently in use. These systems promise to provide significant increases in the effectiveness of aircraft operations and basic aircraft design. Operations can be improved by the increased use of automated functions including the routine use of low visibility automatic landing and by automated monitoring of all aircraft systems which can provide more reliable operations. The basic aerodynamic and structural design of the aircraft can be made more efficient by the use of active control technology as will be discussed in Section 3.4.

Basic electronic components, however, do not have the inherent reliability of many traditional mechanical devices and the basic aircraft structure. In order to give electronic systems the function reliability needed for new flight critical functions, they must be designed to be tolerant of inherent failures. In this area of fault tolerance, aeronautical equipment has a much greater requirement than the great majority of other applications of electronic technology. Thus much of the development of fault tolerant systems must be done specifically for aeronautical applications without being able to fully depend on other more generally supported developments.

Fault tolerant systems are made up of several sub-elements each of which has to have a comparable degree of fault protection for the total system requirements to be met. One basic element in a fault tolerant system is a fault tolerant computer complex. The computer complex must be supplied with data from fault tolerant sensor systems and command inputs and be able to apply outputs to fault tolerant actuators and displays. These computers, sensors, and effectors must be integrated

together into a total system by a fault tolerant data communications system. This entire system must be supported by fault tolerant sources of electrical power and possibly hydraulic power.

Software Implemented Fault Tolerance.

by SRI International & Bendix

The major units of a SIFT system are shown in Figure 3.1.3-1. The system consists of a number of central processing units with associated memories and input/output processors and their associated memories. These processors and memories are connected by a number of redundant buses. The numbers of processors, I/O processors, and buses are variable and depend on both the reliability required and the size of the computational task.

High reliability is achieved by having critical tasks performed in more than one processor. The number of different processors used for each task can be variable as a function of the criticality of the particular task.

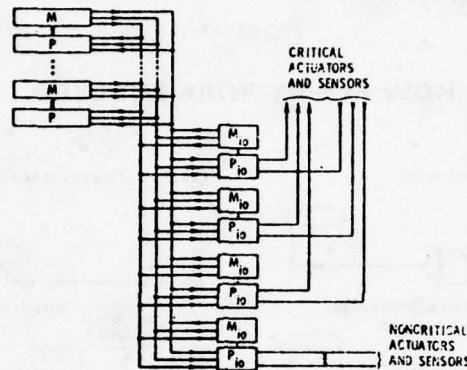


Figure 3.1.3-1 SIFT Configuration

Fault Tolerant Multiprocessor. by C.S. Draper Lab & Collins

A diagram of the FTMP is shown in Figure 3.1.3-2. This system also consists of a number of processor modules, memory modules and I/O access modules connected by a number of buses. Again the number of modules and buses is variable and depends on the reliability and capacity requirements. In this system a memory unit is not associated with a particular processor.

High reliability is accomplished by forming processors and memories into computer and memory triads connected by a triad of busses. In a typical installation there will be a number of triads performing as a multiprocessor. Each member of a single triad executes identical programs in synchronism with the other two members. By noting all the results that appear on the buses, the triad can mask any failure and quickly detect the faulty module. If there is a spare module available, it replaces the faulty one. If there are no spare modules available one triad in the operating set is lost, but two new spares are created. The multiprocessor is designed with sufficient capacity that critical tasks can be performed with the remaining triads.

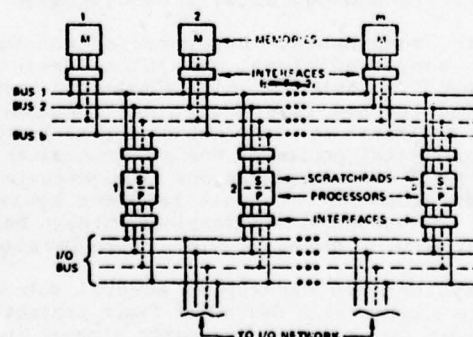


Figure 3.1.3-2 FTMP Configuration

3.2 PROPULSION SYSTEM CONTROLS

3.2.1 BACKGROUND

Most present aircraft turbine engines are controlled by rugged, highly reliable, hydromechanical devices. These mechanical devices consist of cams, linkages, hydraulic servos, etc. They have, over the years, developed a high level of maturity. In fact, the mean-time-to-failure for typical units on civilian aircraft engines is between 20000-30000 hrs. Most of today's commercial engines, however, are relatively simple aero-thermodynamic devices. Fuel metering and possibly some compressor variable geometry are the variables which must be properly manipulated by the controls. On military engines, where supersonic applications are typical, the number of inputs to the engine increases. In fact, on future engines where the design emphasizes high performance for minimum weight, the number of inputs to be controller becomes quite large (5-7).

Experience with the multiplicity of inputs on today's military engines has shown the design of a complete hydromechanical controller to be a difficult, if not nearly impossible task. The computational task to be accomplished accurately and repeatedly by the controller is the reason for the inadequacies of hydromechanical components. As a result, one of the more modern military engines, the P&WA F-100 afterburning turbofan, uses a digital electronic computer as a trim control for a rather complex primary hydromechanical controller. Only with the intelligence of the digital trim control can the F-100 achieve full-rated performance.

Thus, there is at this time a great deal of attention being paid by the aircraft propulsion control industry to determining the technology needs for a more universal applicability of digital electronic turbine engine control.

The current generation of high bypass ratio turbofan engines requires relatively complex systems for internal engine control and protection. Consequently, a proliferation of system components has occurred to earlier engine models. For example, the JT9D employs approximately twice as many control components as does the JT8D. The retention of conventional hydromechanical control methods was a major factor in this trend. Notably, the newly proposed JT10D and CFM56 engines employ hybrid electronic hydromechanical methods.

Several recent programs have successfully addressed the application of electronic controls

PREVIOUS RELATED PROGRAMS

o JT8D-ELECTRONIC PROPULSION CONTROL SYSTEM

JT8D GROUND DEMONSTRATION OF FULL AUTHORITY
DUAL CHANNEL DIGITAL CONTROL

o IPCS-INTEGRATED PROPULSION CONTROL SYSTEM

F-111/TF30 FLIGHT DEMONSTRATION OF FULL AUTHORITY
INLET/ENGINE/AFTERBURNER/NOZZLE
INTEGRATED DIGITAL CONTROL (ONE ENGINE)

o QCSEE-QUIET CLEAN STOL EXPERIMENTAL ENGINE

FULL AUTHORITY DIGITAL CONTROL PRESENTLY BEING
EVALUATED ON GROUND TEST ENGINE

o F100-MULTIVARIABLE CONTROL SYSTEM PROGRAM

F100 GROUND DEMONSTRATION OF AN ADVANCED MULTI-
VARIABLE ENGINE CONTROLLER BASED ON LINEAR-
QUADRATIC REGULATOR DESIGN TECHNIQUES AND
IMPLEMENTED WITH A MICRO PROCESSOR

These programs and other previously conducted studies have produced many affirmative conclusions as to the advantages of electronics in propulsion control systems. Some of the prominent benefits are:

- Increased engine life
- Reduced pilot workload (through improved speed-path controllability)
- Reduced fuel consumption
- Improvement in on-time departures, reduction in unscheduled removals and aborted flights
- Improved stability and failure protection

Several programs which have helped add to the technology base needed for a digital engine control capable of full-authority control of a turbine engine have been either completed or are in progress. Briefly, these are:

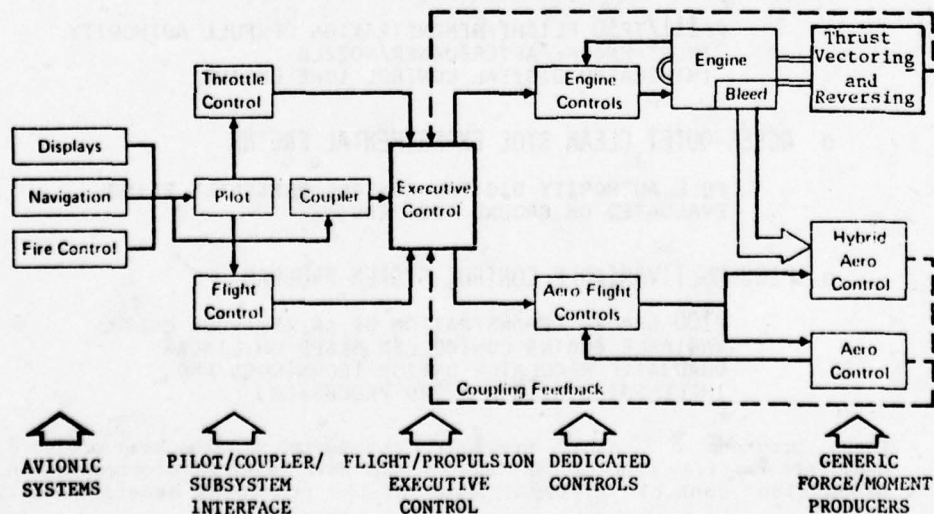
- 1) F-111-IPCS (Integrated Propulsion Control System) program which evaluated in flight a full-authority digital electronic control for one propulsion system of the F-111 E A/C (1975).
- 2) EPCS - Electronic Propulsion Control System which was an industry sponsored effort to develop and evaluate an engine mounted -full-authority controller for an operational civilian aircraft engine. System was demonstrated at sea level.(1975)
- 3) FADEC - Full-Authority Digital Electronic Control is a NAVY sponsored effort to develop an engine control using the very latest state-of-the-art electronic technology. System will be engine mounted and a sea level evaluation and possibly a flight evaluation will be performed. (1978)

The major hurdle for the acceptance of a full-authority digital electronic engine controller is that of reliability.

3.3 INTEGRATED FLIGHT CONTROLS/PROPULSION CONTROLS

Flight Propulsion Control Coupling (FPCC) is part of the trend in advanced aircraft design technology toward more complete interaction and integration of the propulsion system and aircraft controls (Figure 3.3-1) to obtain increased performance. Preliminary studies have been conducted to explore the benefits and design considerations of coupling airframe and propulsion system force producers. From these activities, it was shown that in the design of an advanced aircraft, control of the force production, distribution and management must be addressed collectively, in order to establish the overall weapon system benefits and risks.

FIGURE 3.3-1 FLIGHT/PROPULSION CONTROL INTEGRATION SYSTEM
BLOCK DIAGRAM



3.4 ACTIVE CONTROL SYSTEMS

The term active system has been used to describe systems performing a wide variety of control functions. It will be used here to describe only those systems, utilizing force and moment controls, which are not intended to directly influence the flight path of the vehicle. Active control systems, specifically, are those feedback control systems which sense aircraft motion or structural responses and tend to diminish or limit some variable by introducing counteracting control forces or moments. The functions performed by active controls include the augmentation of the inherent aircraft stability, limiting of maneuver loads, alleviation of gust load, augmentation of the stability of flutter or other elastic modes, modification of load spectra to improve fatigue life, and alleviation of flight station and/or cabin motions to improve ride quality.

FIGURE 3.4-3 NEED AND IMPORTANCE OF ACT TECHNOLOGY AREAS

Technology Areas	ACTIVE CONTROL FUNCTIONS				
	Relaxed Stability	Maneuver/Gust Load Alleviation	Flutter Mode Control	Fatigue Life Improvement	Ride Quality Control
Criteria	* 2/3	3/3	3/3	2/2	2/1
Validation Techniques	3/3	3/3	3/3	2/2	2/2
Data Base	2/3	3/3	3/3	3/2	2/2
Modeling and Analysis/Synthesis Techniques	1/3	2/3	3/3	2/2	2/2
Concepts and Mechanizations	2/3	2/3	3/3	1/1	2/1

* Need/Importance 1 - slight 2 - moderate 3 - great

4.0 CREW STATION TECHNOLOGY

The cockpit displays and controls are undergoing a dramatic change as digital avionics bring more functions and modes, but the control panel remains limited. A strong trend is emerging in crew station technology which is pushing toward multi-function, interactive displays to accommodate the increased functions in the limited panel area.

The increased functional load and attendant multitude of modes are causing a potential problem of learning how to operate the systems in an already overly complex cockpit environment. The most promising system solution to the management of cockpit complexity is the introduction of hierarchical, prompting menu selections for the many functions and modes which must be controlled from the limited panel area. This system technology has been studied and refined by many investigators in the field, but requires considerable additional refinement to assure acceptable operation in a busy IFR environment.

James E. Gorham, testifying for the AIAA, identified the need for someone, NASA or FAA, to undertake the complete redesign of cockpit displays, integrating as many systems as possible to reduce complexity and save weight. James J. Kramer, Associated Administrator of NASA's Office of Aeronautics and Space Technology, stressed the necessary

inclusion of human-factors engineering in the design of the cockpit and proposed the incorporation of advanced electronics to simplify the pilot's task and to improve safety.

The dramatic advances in electronics technology over the past few years that Kramer referred to, coupled with an improved understanding of human factors, provide us with the opportunity to improve greatly the functional range and performance of these future man/machine systems.

4.1 DISPLAYS

Progress is being made in many areas of displays technology which can have significant impact on the design, configuration, and performance of civil crew stations of the future. This display technology has the potential to help declutter the cockpit, reduce pilot workload, enhance flight management, and, thereby, improve operational safety and efficiency. In addition, it has the potential to help reduce the size, weight, and life-cycle cost of cockpit avionics through improved display flexibility, reliability, and maintainability. The basic thrust of many efforts in the crew station technology area is to develop the capabilities for computer-based (or microprocessor-based) electronic displays to replace cluttered arrays of electromechanical instruments. Crew station display concepts which are emerging can be categorized as follows:

- Programmable Electronic Multimode Displays

Electronic display systems, used as primary flight indicators, navigational indicators, engine indicators, and systems status and/or warning indicators. These systems can: (1) present information in pictorial and/or abstract formats which can be more readily interpreted by the pilot; (2) time-share the precious display panel space using multimode presentations; and (3) be reprogramed with new formats when aircraft are assigned new missions or as new sensor and data link information becomes available.

- Advanced Input/Output Techniques

Advanced display media, used in a multifunctional switching capacity, or, in the form of voice synthesis and recognition (VRAS) devices, to reduce the number of single-function switches and, consequently, pilot scan pattern and entry errors.

- Display Data Busing Techniques

Electronic and electro-optical techniques appropriate to multiplexing data between the cockpit equipment bay (display generators) and the cockpit instrument panel (electronic displays) which can reduce the clutter, size, weight, and number of avionics cables, while at the same time implementing methods for display redundancy management.

- Display-Based Automatic Checkout Concepts

Advanced display media and electronics systems which facilitate cockpit avionics systems checkout and maintenance through compatibility with built-in test equipment (BITE), thus increasing the safety of operation.

- Display Modularity and Redundancy

Modular electronic displays which can be physically and electronically interchanged, with formats being determined through display generator programing or selection from a multiplexed data bus. These devices can provide extensive reversionary capability (in the event of display failure) and greatly reduce the spares complement required for maintenance purposes.

4.2 PILOT CONTROLLERS AND DATA ENTRY

Pilot controllers and data entry methods have been a slowly changing part of crew station technology; however, the increase in digital avionics functions implemented in civil aircraft is forcing a change in the traditional approach to controllers and data entry. The traditional approach has been to use dedicated controls and data entry means for each function. As the number of functions and modes proliferate and the amount of cockpit area remains largely fixed, it becomes necessary to devise means to select many functions and modes from an integrated data entry & control center (IDECC).

4.3 FLIGHT SYSTEM MANAGEMENT

Flight management is becoming an important function in digital avionics systems. Flight management includes a number of subfunctions:

- Flight Plan Data Entry
- Avionics Function/Mode Selection
- Data Entry and Display
- Flight System Warning Displays and Annunciators
- Data Base Management
- Aircraft Performance
- Checklists and Other Handbook Data

The complexity of the flight systems management task is growing which requires a break in the traditional approach of dedicated controls and displays for each avionic functions as discussed in section 4.2. The use of multifunction controls and displays demands a new approach using interactive displays which simplify mode selection procedures through the use of computer prompts. The new approach is flexible, conserves panel space, and is easy to learn.

5.0 INTEGRATION AND INTERFACING TECHNOLOGY

A significant departure in avionics systems architecture was initiated by the USAF DAIS which is a distributed system with functions interconnected by means of a serial digital multiplex data bus. The trend toward functionally modular, distributed avionic system architecture has been accelerated by the advent of low cost microprocessor technology. This distributed, functionally modular architecture is now being applied to both airline avionic systems and general aviation avionic systems.

The explosion in airborne information processing capability is continuing. Knowledgeable technologists in very large scale integration (VLSI) circuits predict another two to three orders of magnitude increase in circuit density and gate speed during the next decade. The mass memory technology is providing a bridge between the relatively slow rotating magnetic media storage systems and the very fast but expensive semi conductor memory. This fast, large capacity mass memory provides the means for storing enormous data bases for navigation and flight management functions in the next generation avionic systems.

The use of fiber optics for electronic system interconnect and for multiplex avionic data busses is emerging from the laboratory to the possibility of an economically feasible replacement for copper wire in the next decade. Such fiber optics interfaces will greatly reduce the threat of EMI and indirect lightning effects on digital avionics.

5.1 AVIONICS FUNCTIONAL INTEGRATION

The current trend toward modular digital avionics is greatly simplifying the functional integration problem. Each avionic function is being implemented with a dedicated digital processor(s) with its associated software (or firmware). The functional element receives its required data from the multiplex data bus and supplies its data output to the bus for other functional users.

Historically, mission information requirements have been established along semiautonomous subsystem areas such as flight control, navigation, communication, stores management, weapon delivery, etc. The DAIS approach proposes that the various standard modules be common to all subsystems on an integrated basis. This will not only reduce costs associated with the current proliferation and nonstandardization of modules, but will also provide the opportunity to easily share information between subsystems. This latter feature can enhance mission effectiveness and also provide functional redundancy for increased mission reliability.

The emergence of microprocessors on system integration is an area worthy of increased emphasis. The impact on system partitioning, preprocessing of information, redundancy management, and software structures should have very innovative impacts on future architectures.

5.2 AIRBORNE INFORMATION PROCESSING (DIGITAL COMPUTERS)

PROCESSORS

About five years ago (1973) the first monolithic, LSIC digital processors appeared on the market. The development and use of this type of device, which for obvious reasons was called a microprocessor, has accelerated greatly, and at the present time a wide variety of microprocessors (and a further development, the microcomputer) are available commercially.

At this point, it may be well to define the terms 'processor' and 'computer' as they are used here. A computer is an assembly which contains the following functional elements;

- Arithmetic Logic Unit (ALU)
- Processor control/executive
- Input conditioning
- Output conditioning
- Memory (scratch pad and program)

while a 'processor' commonly encompasses only the first two elements and must be accompanied by additional circuitry which performs the remaining functions. Both types require an external power supply and in many cases a separate clock.

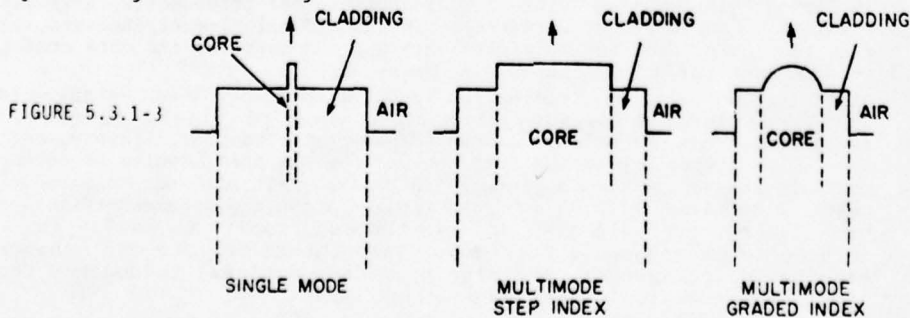
During the past five years there has been continual progress in LSIC technology toward putting more logic and memory elements on one chip to the point that the latest devices are referred to as VLSIC (Very Large Scale Integrated Circuits). The capability of producing more complex chips has resulted in the emergence of the microcomputer mentioned earlier. These microcomputers comprise all the five elements listed above, but to date the inputs and outputs are digital only. Analog to digital, and digital to analog conversions and demodulation must be handled in separate circuitry. Recent design studies of microcomputer-based computers suitable for the military airborne environment indicate that a volume of 300 cu. in. and a weight of 10 lbs. for a computer with a 1-2 microsec. add time and perhaps 8K program memory words is reasonable. Hybrid and/or monolithic converters suitable for many applications such as flight control and inertial navigation computers have recently become available in 8 and 12 bit form and will soon appear in 16 bits. The use of these I/O devices and the latest memory chips (16K RAMS & ROMS) could now result in a total package of approximately 150 cu. in. and 6 lbs.

5.3 INTERSYSTEM COMMUNICATION

5.3.1 FIBER OPTICS

Fiber optics is the technology by which light is coupled into and transmitted from one point via a glass waveguide to another point. The light is generated at the transmitter end by an optical source such as a light emitting diode (LED) or semiconductor laser and detected at the receiving end by a photodiode (pin or avalanche).

Illustrated in figure 5.3.1-3 are the three main types of fiber waveguides and the various index of refraction profiles. The step-index has the lowest bandwidth and the single mode fiber the highest. For avionic applications single fibers of the multi-mode type will probably predominate as light waveguides until Gigahertz bandwidths are required or optical switching techniques become a major requirement in data processing and handling.



5.3.2 BUSSING CONCEPTS

The current trend of distributed computation architecture in avionics systems brought about by the advent of low cost microprocessor technology has given rise to the problem of intra element communications. The intra element communication problem is being solved in today's and the next generation avionic system by the use of data busses.

The general philosophy of the data bus is to place all system parameters or data required by two or more elements of the distributed system on the data bus. The data is transmitted over the bus in a sequential fashion such that each user element may access that data needed. The data sequence must be coded in such a way that each user can determine that a particular data subsequence is that needed by the user. Also it is necessary for some system elements to supply sensed data or processed data to the bus. Also it is necessary for one element of the system to act as a bus controller to control and synchronize the data traffic on the bus.

The data bus normally includes three elements:

- o Physical Transmission Medium
- o Bus Interface Units
- o Bus Supervisor (or Controller)

The transmission medium in the current generation of systems uses conductive paths or wires in a cable for the data transmission. The number of wires in the transmission medium depends upon whether the data is transmitted bit parallel or bit serial. There is a trend toward the use of fiber optics for transmission medium as discussed in section 5.3.1 for future systems.

There are three specifications which cover digital data busses and interconnections. The IEEE 488 is a 16 bit parallel bus which has been considered for use in NASA Ames' Preliminary Candidate Advanced Avionics System Study. The ARINC 419 Spec is a compendium of digital interconnection used by the airlines. The MIL-STD-1553A is a serial digital bus originally adopted by the USAF for the DAIS program which is becoming the standard for military avionics applications.

Harris Semiconductor has also produced a chip which implements a portion of the MIL-STD-1553A bus interface unit (BIU) function. It is expected that in a short time a chip family which implements all of the 1553 BIU functions will be available.

The next step as VLSI and ULSI chips are developed will be to integrate the bus interface unit functions on a microprocessor chip in order to reduce the number of pins on the chip.

5.4 EXTERNAL INTERFERENCE EFFECTS

5.4.1 LIGHTNING EFFECTS ON DIGITAL AVIONICS

Both lightning and static electricity constitute a potentially serious threat to aircraft electrical and electronic subsystems. In particular, advanced digital avionic systems which in the near future are expected to perform a variety of advanced flight and mission critical aircraft functions must be reliably protected against the effects of voltage and current transients due to directly attached or nearby lightning discharges. The increasing use of high resistivity advanced structural materials may complicate these protection requirements.

5.4.2 ELECTROMAGNETIC INTERFERENCE

The control of interference effects on B-1 electronics has been successful using present technology for integration and interfacing.

Suppression or elimination of conducted or radiated interference has been the objective at the interference source, in the coupling path and the receiving circuits. Present technology using filters, shielding, wire twist and shielding, bonding and grounding concepts have been successfully applied as part of the normal design process to meet specified EMI requirements for interference and susceptibility.

One significant design concept used to control EMI was the use of a two (2) wire power distribution system. Power was provided to the using system and load currents were all returned to a single point power ground, thereby eliminating interfering power currents through the vehicle structure. The signal and chassis grounds for each electronic

system used the immediate structure support and interfacing systems used differential signals to isolate signal grounds and limit effects of common mode pulses.

6.0 MILITARY TECHNOLOGY

Military Avionics Technology has an indirect connection to the NASA role which couples through common mechanization elements such as digital processing components and through common system technologies such as modern software development methods.

This SOAS report has included selected military technology areas because of the potential technology transfer from the military to the NASA mission.

7.0 FUNDAMENTAL TECHNOLOGY

The fundamental technologies form the basis for the other technologies by providing the analytical tools, the software development methods, and the emerging new components. This technology area covers Information and Control Theory, Device Technology, Fluidics, System Analysis and Software Methodology.

7.1 INFORMATION AND CONTROL THEORY

The theoretical foundation for information and control theory as used in modern control today is approximately 20 years old. The mathematical formulations were considerably more sophisticated than the classical control methods that had been used previously. As a result there was a longer learning time before the system application implications were fully understood and there was an adequately educated community to accept the modern approach. In some cases, the classical designs have continued to be the designs of choice, appropriately so because of their simple structure, and in many cases, easier accessibility for checkout, and trouble shooting in the field. Inevitably, we have progressed in our challenge to control more complicated systems so that we must rely on the formalism on the multi-input multi-output control theory.

7.1.1 OPTIMAL CONTROL

NASA should investigate methods of designing robust optimal control laws. These control laws would either be passively insensitive to plant variation or would actively adapt to variations. Passive insensitive controllers may be more practical than the active adaptive methods. In general, filters and state estimators, essential components of a complete optimal feedback control scheme, tend to be more sensitive to unmodeled plant variation than state feedback controllers. This phenomena should be studied. Applications which will require optimal controllers which function well despite plant uncertainties include:

- (a) control of flexible structures (e.g. solar power satellites)
- (b) Light-wing-loading STOL ride quality control
- (c) reliable automatic landing under operational variations in system models
- (d) helicopter handling quality (e.g. vibration, gust, response time) and fuel efficiency improvement
- (e) Extended flight envelope autopilots

7.1.2 KALMAN FILTER AND STATE ESTIMATION THEORY

Implementation of this technology presents the greatest technical opportunity at this time. NASA should investigate the application of modern identification techniques to the task of optimizing load and ride control system performance. The ability to correctly predict structural mode shapes during the design of an airplane is presently poor. Identification techniques can be used to generate high fidelity airload/structural models from flight test measurements. These models could then be used to fine tune the load/ride control system. The key

7.1.3 DIRECT DIGITAL SYNTHESIS

The trend to digital mechanizations has fostered a re-examination and extension of techniques for discrete systems analysis and synthesis. Most of the control problems of interest involve a continuous controlled element and a hybrid controller which may contain both continuous and discrete elements. Since a major portion of the system may be continuous, and because many system design criteria and procedures have been developed for continuous systems, one very popular approach is some form of emulation.

7.2 DEVICE TECHNOLOGY PROJECTIONS

7.2.1 ELECTRONIC DEVICES

This section presents a review of the current status and projection of the trends of various technologies to 1985-1988. A 3- to 5- year time lag is assumed at that time to incorporate this new technology in production hardware.

7.2.1.1 Silicon Semiconductor Technology

Circuit Technology

Advances in semiconductor state of the art over the last 5 years have been impressive. The basis for much of the recent progress in integrated circuit (IC) technology has been advances in photolithography, semiconductor cell isolation, and ion-implantation processing schemes. Semiconductor chips with the equivalent of 80,000 transistors are available as off-the-shelf items, and microcomputers, microprocessors, programmable hand-held calculators, and complex memory chips are readily available. Development of newer methods, such as electron-beam and ion-beam implantation pattern generation, promise more improvement. To prevent mask damage due to contact printing, efforts are also underway in projection printing and 'near-contact' printing. Feeding these developments is an array of technologies remarkable for their diversity and ability to enhance circuit performance.

7.2.1.3 Gallium Arsenide Technology

Gallium Arsenide is an emerging technology which promises to bring a further revolution to circuit technology beyond that possible with silicon technology. The basic physical characteristics of GaAs provide an electron mobility (μ) five times that of Si which means GaAs has a great potential for smaller power-delay products than for equivalent Si devices. Furthermore, GaAs has the potential of operating above 400 deg Celsius and will sustain greater nuclear hardening than with Si (10 exp. 15 to 10 exp. 16 neutrons per square cm. and 10 exp. 5 to 10 exp. 6 rads of gamma radiation). Currently there are three GaAs technologies being exploited for circuit application:

- Enhanced Junction Field Effect Transistors (E-JFET)
- Metal Semiconductor Field Effect Transistors (MESFET)
- Transfer Electron Devices (TED)

7.3 FLUIDICS

NASA's current effort in fluidic technology has been directed toward certain carefully chosen avionics applications in which fluidic devices would appear to offer some real advantage over more conventional systems in terms of reliability, economy and simplicity.

7.4 SYSTEMS ANALYSIS

A great many facets of systems analysis have already been covered under other headings in previous sections of this report. For many guidance and control and other avionics purposes the available system analysis theories are completely adequate.

However, for the stability and performance analysis of complex high-dimension nonlinear and time-varying systems, the support tools of computation and data presentation are, at best, treated as separate special cases. Approaches which can be used to classify, arrange, and bring order, clarify, and perspective to vast masses of data are simply not available. Thus, we can analyze almost anything, yet can appreciate and fully comprehend very little. To address these problems requires a hierarchical approach at the one extreme and a limited case/simplified approximation at the other.

7.5 AVIONICS SOFTWARE

The trend in avionics is to increase the number of functions implemented in digital processors. As a result, the avionics software complexity has continued to increase. The trend toward distributed processing architecture is alleviating the problems of large software programs by dividing both the hardware and software into manageable modules.

The use of large real time software programs in large central computer complexes as exemplified by the F111 Mark II avionics and the Shuttle avionics gives rise to difficult software validation problems. However, this type of architecture is diminishing in its applications.

The proportion of software costs to hardware costs is growing with each new generation system as low cost digital microprocessor/memory technology is introduced.

Most current generation avionic and control systems have utilized assembly language programming for the software. There is a strong trend toward the use of higher order language, (HOL) for software development. The use of HOL increases the programming productivity considerably and reduces both the initial software development costs as well as software maintenance costs.

Software almost always takes longer to develop and costs more than originally estimated. Furthermore, it is difficult to obtain meaningful completion status information, e.g., there is the 90% done but 90% left to be done syndrome in attempting to assess the schedule status.

Software is unreliable. Software often fails to meet the specifications and contains undetected coding errors which appear later after more thorough system testing has taken place. Software causes limitations on system performance.

Software is fragile in that a validated software program is vulnerable to modifications often in an unpredictable manner. Software often has a poor tolerance to off nominal use of the system, particularly, in regions which were not tested during validation.

DOD has an interim list of approved HOLs (ref. 126) which include:

- FORTRAN IV, ANSI version
- J3 JOVIAL
- J73 JOVIAL
- TACPUL
- SPL-1
- CMS 2
- ANSI COROL

LIST OF REFERENCES

1. GAO report, "Navigation Planning -- need for new direction," LCD-77-909, March 1978
 2. R. H. Pursel, Jack D. Edmonds, "A Flight Investigation of System Accuracies and Operational Capabilities of an Air Transport Area Navigation System," FAA-RD-76-32 (N76-29205), May 1976.
 3. K. E. Duning, et al, "Curved Approach Path Study," FAA-RD-73-143, April 1973.
 4. N. B. Hemesath, et al, "Three and Four Dimensional Area Navigation Study," FAA-RD-74-150, June 1974.
 5. J. M. H. Bruckner, et al, "3D/4D Area Navigation System Design Development and Implementation," FAA-RD-77-79, I, June 1977.
 6. F. Neuman and H. Q. Lee, "Flight Experience with Aircraft Time of Arrival Control," Journal of Aircraft, June 1977.
 7. W. R. Wehrsend, et al, "Description and Flight Performance of Two Systems for Two-Segment Approach," AIAA paper 74-980, Aug. 1974.
 8. Robert R. Ropelewski, "Competing MLS Systems Complete Brussels Tests," Aviation Week & Space Technology, February 13, 1978.
- Department of Transportation, Department of Defense, and National Aeronautical and Space Administration, "National Plan for Development of the Microwave Landing System," Washington, D.C., July 1971.
10. Federal Aviation Administration, "Time Reference Scanning Beam Microwave Landing System, Introduction and Summary," Section 1.0, U.S. Proposal for a New Non-Visual Precision Approach and Guidance System for International Civil Aviation, October 6, 1977.
 11. Ibid., p. 1-12.

NOTE: The NASA Contractor Report CR 159050 entitled, "State of the Art Survey of Technologies" edited by Dr. R.K.Smyth, dated October 1978, contains 132 references. Limited space did not permit including all references in this paper.

A FLIGHT CONTROL SYSTEM USING THE DAIS ARCHITECTURE

by
 A. P. De Thomas
 Major R. A. Hendrix
 Air Force Flight Dynamics Laboratory
 Flight Control Division
 Wright-Patterson Air Force Base, OH

SUMMARY

This paper summarizes work being accomplished in the development of a digital flight control system simulation at the Air Force Flight Dynamics Laboratory. The principle purpose of this activity is to provide the capability to examine advanced integrated control architectures which will increase system performance and availability.

1. INTRODUCTION

The purpose of this paper is to discuss an engineering tool under development at the Air Force Flight Dynamics Laboratory (AFFDL) which is being used to investigate certain critical issues related to advanced digital flight control systems. This simulation will allow the capability to investigate near term issues such as multiplexing, interfaces with other avionics functions, and the structuring of software. In broader terms, this system will be used to investigate more advanced architectures which consider the functional integration of flight control with other aircraft functions.

2. DISCUSSION

As digital fly-by-wire techniques emerged, an expanded technological base developed which allowed the application of advanced flight control concepts. Through the use of active controls, an improvement in range and ride quality was demonstrated and the addition of multimode control laws showed and enhanced maneuver and precision tracking capability through the use of task tailoring. Digital technology also shows a combined improvement in safety, reliability, maintainability, and life-cycle costs. These latter improvements are attributable to the great strides achieved in digital integrated electronics which made practical the application of redundancy techniques, built-in-test capability, and hardware standardization strategies.

The Digital Avionics Information System (DAIS) Advanced Development Program was established by the Air Force as an approach for reducing the life-cycle-costs of systems. As will be discussed later, the DAIS system provides a flexible information transfer network which allows the integration or sharing of data by all subsystem elements.

The next generation of flight control is considering integration at a higher system level. In this approach, the system functions are integrated through software, and sensors and other hardware components are shared so that mission effectiveness is improved along with an increase in system and mission reliability. The types of systems to be integrated with the flight control system are weapon deliver, propulsion, navigation, threat avoidance, command-communication-control, and systems management. The net effect is to reduce system costs and increase system reliability and performance because of the blending of functions and the sharing of information and hardware elements within the system. The integrated system will also unload the pilot of routine tasks and place him in the role of system manager versus system operator. This would be accomplished by integration of much of the data normally presented to the pilot and presenting him with directly usable command information. Another function of the integrated system will be the development of an exhaustive self-test capability which can isolate failures to the card level with the on-board computer system and then reconfigure the system to allow continuation of the mission. The goal is to eliminate complex ground checkout equipment and to increase system availability. By proper application of this technology, the survivability of the system can also be greatly enhanced.

To achieve these goals, the emerging very high density integrated electronics is seen as a key element which will bring these concepts to fruition. However, before this can be accomplished, the development of flexible analytical and simulation tools must be established to investigate the architectural alternatives and trades between hardware and software.

3. THE DAIS SYSTEM

The DAIS system concept proposes the integration of system subfunctions through the use of common data transfer and processing elements. Information within the system can now be easily shared among the subsystems so that it can be used in a more optimal manner. By using commonality throughout the system, the system can be easily reconfigured for use in a wide variety of aircraft types and missions. The overall goal is to reduce the life-cycle-cost of systems.

The DAIS program objective is to demonstrate the DAIS concept through use of a

specific system architecture; namely, a night, all weather close air support mission based on the A-7D aircraft. The architecture is built with a set of standardized hardware and software modules and has the flexibility to be easily reconfigured for evaluation of alternate approaches. The modules include processors, multiplex data bus and interfaces, controls and displays, and software.

The modules are being assembled into a hot bench at the AF Avionics Laboratory and into a Flight Simulation Facility at AFFDL. This latter simulation will be utilized to evaluate flight control system performance, interaction of flight control and avionics, and the pilot interface with the system.

The overall DAIS architecture is shown in Figure 1. Because of differing redundancy requirements for safety-of-flight integrity, the system is partitioned into an 1) avionics section which handles the traditional avionics functions, such as navigation, stores management, weapon delivery, communications and 2) a flight control section which handles the inner-loop stability functions and other information deemed necessary for safety-of-flight integrity.

Data distribution within the system is via multiplex data bus, specifically MIL-STD-1553A. This system uses time-division multiplex (TDM) with a 1 MHz manchester bi-phase coded signal. The remote terminals (RT) provide the interface with the subsystems and are configured with standard modules. The bus controller interface unit (BCIU) is either programmable or controlled by the processor. The bus controller commands the transfer of data on the bus and also performs a bus monitoring function.

The processor is designated the AN/AYK-15 and is a general purpose 16 bit machine which supports higher order language (HOL) programming. Currently, the JOVIAL J-73/I language is being utilized. The functions performed by the processor are: master executive, local executive, applications program, and system reconfiguration. The processor speed is 400KOPS, has floating point capability, is micro-programmable, and can be configured with up to 64K of memory.

The control and display system interfaces with the avionics data buses and drives cathode-ray tubes (CRT) and control panels in the cockpit. The main features of this system are the modular programmable display generators (MPDG) and display switch memory unit (DSMU) which respectively generates symbology and allows transfer to the CRT of interest. The system can produce either raster or stroke written displays. The raster displays operate at either 525 or 875 lines at a 30 frame per second rate.

The DAIS software consists of mission software, non-real time software, and real time support software. The mission software performs the operational flight programs such as flight control, navigation, weapon delivery, bus control, etc. The non-real time software supports the design, development, verification, and management of the mission software. The real time software represents the software required to operate, monitor, and evaluate the system. It consists of sensor simulations, data collection and analysis, flight simulation, etc. The software is written in HOL.

The avionics system is being implemented with a dual-channel multiplex system with standby redundancy while the flight control system is quad-redundant with all channels active. Both systems utilize the same hardware and are integrated through an asynchronous interface which allows data transfer between systems.

Although the facility is presently being implemented with quad-redundancy, the capability exists to investigate differing architectures and redundancy management schemes, for example, triplex or duplex systems with a multiprocessor arrangement. The inherent flexibility built into the simulation will allow support of advanced integrated control programs within AFFDL as discussed in Section 2.

The avionics portion of the DAIS system employs a federated computer architecture. In this architecture, each processor has its own dedicated memory, and communication among processors and subsystems is by data transfer on the data bus only.

One processor is designated as master with backup master executive residing in the remaining processors. Each processor also contains a local executive. Redundancy is accomplished by the capability of another processor to assume master status upon detection of no bus activity for a designated period of time. Mission essential software can also be reloaded over the bus, from a mass memory unit, in case of processor failure. The executive software is configured to accommodate from one to several processors so that system reconfiguration is easily accomplished.

Figure 2 shows a view of the DAIS-configured cockpit. The cockpit is implemented with cathode ray tubes to provide a head-up display (HUD), vertical situation display (VSD), horizontal situation display (HSD), and two (2) multipurpose displays (MPDs). The cockpit also contains a master mode control panel, and an Integrated Multifunction Keyboard (IMFK). Only that information which is required for a particular flight mode, as brought up by the Master Mode Control Panel, is presented to the pilot. Control of subsystems, such as communications, navigation, stores management, and flight control system mode changes, etc. are accomplished through the IMFK. Display redundancy is achieved by the capability to switch displays among the various CRTs. The shared controls and displays provide flexibility in accommodating changes in missions and an increased reliability due to redundancy of the hardware. Backup electromechanical instruments

for critical flight parameters are provided in case of total avionic system failure.

4. FLIGHT CONTROL SYSTEM

AFFDL is developing the digital flight control system simulation to verify that the DAIS concepts of a system using standardized hardware and software modules, time division multiplex communication, and high order language are applicable to the aircraft flight control task. The system uses the DAIS developed hardware and software modules discussed in Section 3.

A single channel, as shown in Figure 3, consists of a processor, bus controller, and three remote terminals. The remote terminals provide all input and output for the flight control system. Sensor inputs are provided via direct analog input interfaces, which are centralized analog-to-digital converters in each remote terminal. Discrete inputs (mode and trim) are interfaced with the system via discrete input interface modules. Actuator output commands are provided through the appropriate output interface modules. Additionally, a serial digital channel is provided for interface with the avionics multiplex data bus. This channel passes pilot mode requests, computer mode engagements, attitude, and air data information. The processor executes the flight control algorithms in addition to handling the multiplex data bus traffic. The executive is presently organized in a sequential, non-priority form. The flight control system algorithms are modeled after the A-7D DIGITAC multimode control laws.

The flight control system is a quad-redundant, asynchronous implementation, with each of the four channels performing identical functions, Figure 4. The decision for quad-redundancy was based on achieving a failure rate of less than 1 catastrophic failure in 10^7 operating hours for 2-hour flights. It was also decided that no sophisticated techniques be employed for failure detection. The sensor inputs are cross strapped at the remote terminals and the information is sent to the flight control processors over the redundant multiplex busses. Each processor receives four sets of sensor data and selects the lower median value of each parameter. This first voting plane, for sensor signal select, is implemented in software.

Each processor independently computes the appropriate flight control algorithms for the current mode and transmits control surface commands over the multiplex bus to the remote terminals. There then results a quad-redundant set of control surface commands which is processed by digital hardware voter monitors, the second voting plane.

The digital hardware voter/monitors (DHVM) select an output command from the set of commands produced by the processors. It does this by selecting the algebraically lower median of four acceptable signals, the median of three acceptable signals, or the lower of two acceptable signals; its output is a pulse-width modulated (PWM) signal representing the selected input. The selected signal is used for comparison monitoring to detect failed or out of tolerance channels.

The DHVMs are quad-redundant, with each voter providing an input to one channel of a four channel, forced summed hydraulic actuator simulator, which is the third and final voting plane.

The concept of standard hardware modules is being tested by using processors, BCIUS and RTs in the FCS which are identical to those in the avionics section, with the exception that the F/C processors will probably require less memory. The advantage of using the same equipment for different functions is a great cost benefit; however, some loss of efficiency can be expected because the hardware is not optimized to perform a specific task as in the past. This does not present a severe problem since the speed of digital hardware is increasing while costs are rapidly declining.

Time-division multiplexing is used for all input/output to the F/C processors. This method of interfacing signals simplifies the interconnection of hardware elements within the system and thus increases system reliability. It also provides a convenient means of interfacing flight control with other on-board functions. Studies are being directed as to how multiplexing can be utilized in an effective manner so that performance and flight integrity are maintained.

JOVIAL (J-73/I) will be used for programming the F/C operational flight program (OFF). This language more readily lends itself to structured programming procedures than assembly language. Almost all current work on digital flight control is using assembly language and there are concerns about efficiency (speed and size) and reliability of code produced by HOL compilers. The DAIS FEF will allow us to evaluate the operation of an HOL-produced FCS OFF in a realistic simulation. The use of HOL in F/C software development could provide cost benefits in the production and maintenance of digital flight control systems.

The simulation system will initially be tested with the A-7D DIGITAC control laws. Since this system was flight tested, a good baseline of data is available for comparison. The control laws have been written in the MACRO-11 assembly language and tested on a PDP-11/40 minicomputer in the facility. This has provided an additional baseline system to be used in the facility. The control laws are currently being coded in J-73/I and will be tested in the latter part of 1979.

The most obvious approach to redundancy management of multiple channel flight

control systems is to run the channels synchronously. If the various processors operate on the same data with the same algorithms, the outputs should be identical, and selection of a failed channel should be straight forward. However, it is very difficult to develop a synchronizing process which does not require a high reliability hardware controller. The controller must be high reliability because it introduces a potential single failure point.

An asynchronous system does not require the hardware and software interfaces to synchronize the channels. The hardware and software complexity is thus reduced, but at the cost of inherent differences between the outputs of the channels. These inherent differences make it more difficult to determine when a channel has gone out of tolerance; the trip level in the comparison monitors must be large enough to avoid nuisance trips resulting from normal skew between channels, yet small enough to quickly detect a failed channel. Studies have shown that the errors are no worse than those encountered in analog systems with component tolerance errors.

Since the integrators will have different inputs and sampling times, they will tend to slowly drift apart when the system is operated asynchronously. This problem can be overcome by exchanging integrator outputs across the flight control channels as shown in Figure 5. The integrator outputs are then voted for input to the next integration iteration. If an interconnection fails, its output is declared failed and the information is coupled on a second order basis through the remaining interconnections. Mode engagement discretes are also passed across the interchannel interface to verify that all channels are engaged in the same mode. Mode engagements are voted to prevent channels from entering different modes and falsely failing a channel. An asynchronous communicator, a standard serial digital interface card, has been designed and tested. It contains a buffer memory so that the sending channel can transmit data and the receiving channel can read data independently, each on its own schedule.

5. AVIONICS INTERFACE

The Flight Engineering Facility being developed at AFFDL is shown in Figure 6. The facility consists of the quad-redundant flight control system, avionics system, and support system. The avionics system is a simplified version of that discussed in Section 3. Its purpose is to drive the cockpit controls and displays and to present a realistic interface to the flight control system. Because of the reduced reliability of the avionics system, it must be designed so that a failure will not propagate to the FCS and create a risk of aircraft loss.

The pilot controls, stick and rudder are interfaced to the forward RT in the flight control system along with a dedicated annunciator panel. This panel is a dedicated advisory/caution display which gives the pilot an indication of a failure in a particular axis and the type unit which has failed. Reset can be attempted by pushing buttons labeled pitch, roll, and yaw. Reset simply clears the mistrack counter in the voter/monitor. If the channel has a real failure, the mistrack counter will immediately exceed the limit and trip out again. If a channel has a nuisance trip out (i.e., due to a temporary transient), it will not refail immediately.

For effective flight management there must be an exchange of information between the avionic and flight control systems, but the interface must be designed to isolate failures between the systems. Two avionics/flight control interface units are being developed for the Flight Engineering Facility. One is a serial input/output buffer memory unit similar to what is shown in Figure 5, and the other is a microprocessor controlled serial unit which is able to simplify the interfaces in the avionics and flight control systems.

6. SUPPORT HARDWARE

A pictorial layout of the facility is shown in Figure 7. Shown is the DAIS hardware, cockpit and test operators console, support computers, data link to a DEC-10 system at AFAL, and interfaces to a motion base cockpit, hydraulics test stand, and terrain board.

Four PDP-11 minicomputers support the Flight Engineering Facility with various commercial peripherals and several in-house designed and built units for special applications.

The PDP-11s perform the functions of aerodynamic model, simulation monitor and control, data formatting, modeling, and data collection and reduction. Special purpose equipment is used to monitor data bus traffic, control the skew between the four asynchronous channels, and act as interfaces between the processors.

Data bus traffic monitoring is to be accomplished with the Bus Monitor Unit, which is connected between the six multiplex busses and the Performance Monitor Control and Processor. It can be programmed to detect and record predetermined message traffic for later analysis.

The FEF Skew Control Unit, under the control of the Performance Monitor and Control processor, controls the timing of the F/C processors so that the effects of asynchronous operation can be studied and controlled. The Skew Control Units provides an ability to synchronize the channels for tests of alternative FCS architectures.

Although the total facility has not yet been completed, the minicomputers, other commercial equipment and in-house developed hardware have been utilized to conduct several studies.

For example, a dual multiplex system was tested and showed no degradation over its hardwired equivalent. The flight control/avionics interface, using a buffer memory was tested to examine the flow of data between both systems and determine if there were any serious timing problems between the asynchronous busses. A microprocessor interface is currently under development and will be examined to determine if there are any significant improvements in performance.

Three pilot-in-the-loop tests were accomplished to determine the effectiveness of the pilot interface with the system. These tests concentrated on use of multifunction controls, display symbology, and effects of failure modes.

7. CONCLUSION

The Air Force Flight Dynamics Laboratory is developing a Digital Flight Control System using the DAIS concepts of standard hardware modules, time division multiplexing, and high order programming language. The purpose is to demonstrate the validity of these concepts applied to Digital Flight Control Systems. In order to test this Flight Control System in a realistic real-time simulation, a Flight Engineering Facility is being assembled. In addition to its primary purpose of providing a realistic environment for the DAIS Flight Control System, the facility will aid in the investigation of other considerations being raised by advancing Digital Flight Control technology. The trade-off between synchronous or asynchronous operation will be examined, as well as, the partitioning between the flight control and avionics systems. Other areas to be investigated are alternate redundancy management techniques and modular, structured software for flight control. The facility will have the flexibility to emulate diverse system architectures and will be a valuable tool for testing integrated control concepts which are now becoming practical.

8. REFERENCES

- (1). Anthony P. De Thomas and Captain George M. Lacy: Guidance and Control Conference, "A Flight Control System Using the DAIS Architecture," 8-10 August 1977, AIAA-77-1100.
- (2). AFFDL Specification PF 000 000, Flight Engineering Facility Program Plan, 15 April 1977.
- (3). AFFDL Specification SF 000 100, System Specification for the DAIS Flight Engineering Facility, 6 January 1978.
- (4). Texas Instruments, Inc., "Digital Hardware Voter/Monitor Breadboard Development and Evaluation," July 1974, AFFDL-TR-74-94.
- (5). Harry W. Schreadley: "Evaluation of a Digital Hardware Voter/Monitor," February 1977, AFFDL-TR-77-30.
- (6). Captain Vincent J. Darcy and Charles R. Slivinsky: "Analysis of Inherent Errors in Asynchronous Digital Flight Control System," October 1975, AFFDL-TR-76-16.
- (7). AFFDL Specification SF 160 500, Computer Program Development Specification for AFFDL/DAIS Flight Control Software, 31 December 1978.
- (8). Lieutenant Thomas W. Johnson: Proceedings of the IEEE 1976, "Design of a Digital Flight Control System Using Area Multiplexing," May 1976, pp 403-410.
- (9). Lieutenant Thomas W. Johnson: IEEE 1978 National Aerospace and Electronics Conference Proceedings, "A Qualitative Analysis of Redundant Asynchronous Operation," May 1978, Vol I, pp 83-90.
- (10). MIL-STD-1553A, Aircraft Internal Time Division Command/Response Multiplex Data Bus, 30 April 1975.
- (11). TRW Defense and Space Systems Group, Inc., "Digital Avionics Information System (DAIS): Development and Demonstration," March 1979, AFAL-TR-79-1027.

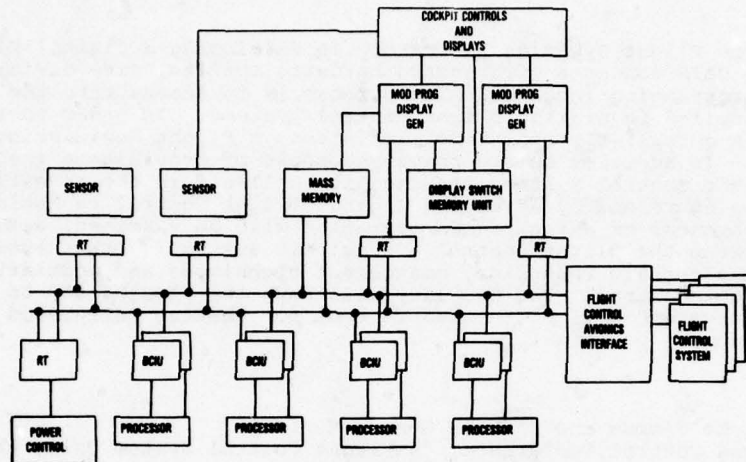


FIGURE 1 DAIS ARCHITECTURE

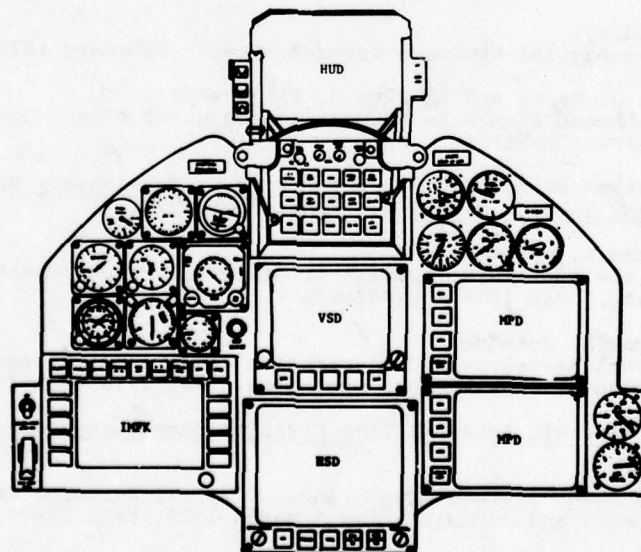


FIGURE 2 DAIS COCKPIT

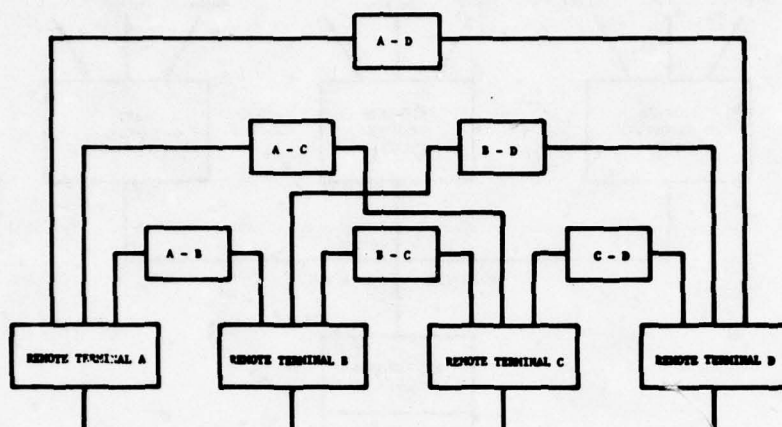


FIGURE 5 INTERCHANNEL INTERFACE

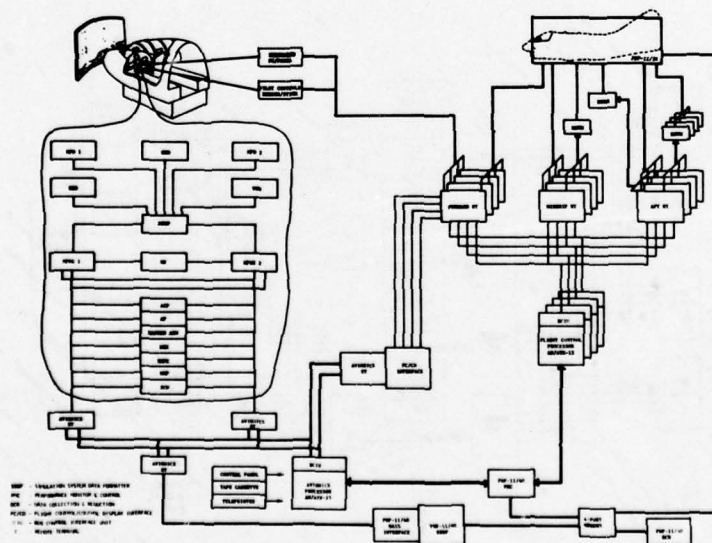


FIGURE 6 FLIGHT ENGINEERING FACILITY

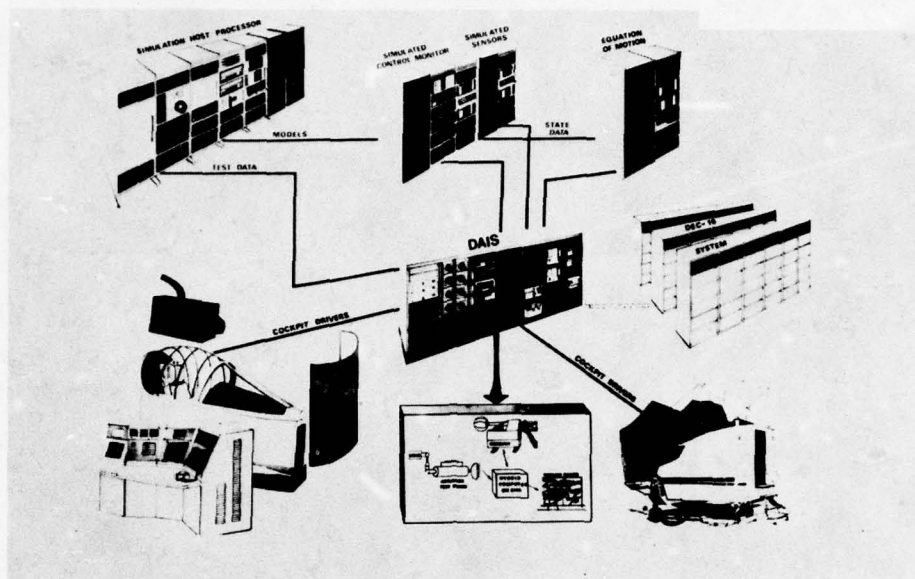


FIGURE 7 FACILITY PICTORIAL LAYOUT

TRENDS IN DIGITAL DATA PROCESSING AND SYSTEM ARCHITECTURE

by

Dr A. A. Callaway

Flight Systems Department
 Royal Aircraft Establishment
 Farnborough, Hampshire
 England

1 INTRODUCTION

The techniques discussed in this paper result from studies currently being carried out at RAE into the utilisation of airborne digital computers and methods for their integration into digital avionic systems. The paper, therefore, is not particularly oriented towards guidance and control as such, but is a rather more general consideration of the avionics application. It is intended to provide an overview of the digital systems scene, and to act as an introduction to some of the techniques which will be discussed during the course of the Symposium.

In our avionic systems, over the past ten years or so, we have seen a massive increase in the utilization of digital techniques in general, and computational elements in particular, as the complexity of the task which the airborne system is called upon to undertake increases. The more capability which we are afforded, the more we will require, and systems will tend to become ever less understandable and manageable unless discipline is applied in their design and realisation.

In order to illustrate this trend, we shall briefly review the architecture of two aircraft systems, one designed in the 1960s and one in the 1970s, and consider the growth in complexity in terms of two factors: the total flow of data between the subsystems which form the elements of the system, and the total volume of the computing task in terms of the number of words of program required.

We shall then consider techniques which may assist in alleviating the growing complexity. For example: design management aids, such as requirement statement languages may have an important role to play; architectural considerations, such as multiplex data busses and distributed processing, and software techniques, such as high level languages, MASCO and structured programming, are all demonstrating that new approaches are important when planning for the future.

2 TRENDS IN SYSTEM COMPLEXITY

The Jaguar is a typical aircraft of the 1960s and its system design and realisation very much reflect 60s technology. The system configuration is of the network type, where each unit which needs to communicate with another is directly connected. There is a diversity of signalling types - voltage analogue, synchro, discrete, encoded discrete and parallel digital - so there are many interconnecting wires, even though there are less than 300 data quantities being interchanged.

The system is also of the central computer type, employing a single computer to provide the system intelligence. The memory size for this processor is of the order of 8000 words, and the effort required to produce the program, written entirely in assembly code, was probably less than 40 man years.

When we move on to the Tornado, we see an aircraft system which incorporates much more digital technology. The system configuration is still of the network type, and still comprises a mixture of signalling types, although there is a significant increase in the proportion of digital transmission, which is principally in serial form and 'information multiplexed', which reduces the number of interconnecting wires. By information multiplexed we mean that if unit A wishes to send data to unit B, it still has its own private connection, but it can send a number of different items of data along the same channel.

The total system data flow requirements, however, represent an increase by a factor of about three over the Jaguar system and, therefore, the resulting system configuration is much more complex. The system contains digital processing in a number of areas - such as embedded processing in the air data computer, inertial navigator and head-up display - but this is dominated by the large central processing task. In total, there is probably of the order of 50000 words of program - still in assembly code - the generation of which has probably consumed more than 300 man years of effort.

The lessons to be learned from the obvious trends in these two programs are that if future requirements are to grow more and more complex, requiring still more digital processing, then the system data flow requirements must not be allowed to grow in proportion, the system configuration must move away from the point-to-point private line concept, and the processing must be organised into more manageable distributed packages.

Technology affords the means to encapsulate the complexity required by more intelligent data distribution, and to distribute processing according to the needs of the system, and this is one of the most important trends in digital systems in the future.

3 REQUIREMENT SPECIFICATION

However systems are configured, they grow ever more complex as to their purpose and requirements. There is more complexity in translating customers' requirements into system designs and in testing these designs against the original requirements. There is more flexibility in determining partitioning strategies and resultant data flow patterns, and there are problems in appreciating the effect of changing requirements. The next topic to consider, then, is whether digital technology has anything to offer in this overall system design area, and here we are considering the application of the computer not as part of the system but as a design and management tool.

One of the initial problems to face in the conception of a new system is the language barrier at various levels of expertise. Consider, for example, that between the customer and designer. The customer knows what the system is required to do but is not necessarily versed in the intricacies of digital systems, and so may find it hard to appreciate what is possible and how to express the requirement in terms which the designer can understand without being ambiguous and insufficiently detailed. The designer then may have no in-depth knowledge of the task or the requirement, but is left with considerable freedom of interpretation because of the lack of rigour in the customers' specification.

This may be an extreme example, but the fact remains that such communication gaps may exist at several levels in a design exercise, and there is no doubt that a common, formal and rigorous method of expressing system requirements can aid enormously in the communication problem and can facilitate the sort of iteration which is clearly necessary when tuning the overall system design and partitioning strategies.

A requirement statement language is a formal method of expressing a design requirement which is then processable by computer and amenable to a number of analyses. The types of checking which can be done may include the following.

- (i) Conformance. Using traditional design techniques, it is difficult to demonstrate that the system design completely conforms to the customers' requirements. Formalising the manner in which the requirement and the design are expressed would facilitate the automatic checking of conformance.
- (ii) Consistency. Inconsistency in a complex design can easily occur, through lack of total visibility. Typical examples may include the production of data by one process of inadequate precision for use in another, multiple generation of data, data required by a process and not supplied by any other, inconsistent data rate requirements, and so on. Automatic design checking can infallibly spot such inconsistencies.
- (iii) Completeness. Completeness of a design assumes all requirements, explicit and implicit, of the customer have been met. If this is not so, then penalties in time and cost escalation or in-service failure will inevitably accrue. Again, formalism of expression aids automatic completeness checking.
- (iv) Feasibility. This means specifying timescale and system characteristics which are capable of being met. The early design stage is the best time to confirm this, and is also the most difficult without the type of detailed system overview the automatic technique requires.
- (v) Trade-off studies. Effects of varying the system partitioning constraints can more readily be studied and changes incorporated, facilitating the customer-designer iterations which are so important.

To sum up, then; in the type of distributed system towards which we are moving, it is important to have a rigorous and unambiguous task description at an early design stage in order to assess such things as system data flow and processing requirements. The total system architecture needs investigation at an earlier stage than the individual subsystem requirements, which can then be hierarchically derived. It is only out of the total system breakdown that such things as system partitioning and data flow patterns can be specified, leading in turn to the subsystem requirements, such as software and hardware specification, data bandwidth, iteration rates, accuracy and precision, and so on.

In order to alleviate the complexity of such an approach, it is clear that use can be made of automated design analysis tools and data-base management aids. It is also likely that such aids will be available to assist in the resource management, as well as design management, throughout the development of such systems. A number of techniques have been developed to undertake such tasks, such as ISDOS, REVS, etc. None has yet been applied to an avionics system project in this country but they will undoubtedly be strongly considered for future programs.

4 THE DIGITAL DATA BUS

One of the most significant advances in methods of system configuration is the adoption of the multiplex data bus for the majority of system data transfers. In adopting a multiplex data bus (mux bus) philosophy, one is trading complexity of interconnection and actual system realisation with complexity of the terminal equipment in each of the subsystems. In the previous network connected systems, where each subsystem has its own private connection to any other with which it needs to communicate, the terminal circuitry does not of necessity need to be of high intelligence, except, perhaps at a focal point of data such as a central computer.

When one moves to a mux bus system where, in principle, all subsystems are connected to a single highway, higher intelligence is needed in the terminal circuitry because all subsystems are now taking part in intelligent conversations. The significant point, however, is that this is manageable complexity since it has been reduced from a configuration problem to a logic problem which is amenable to the advantages offered by LSI technology. In other words, if subsystem designers can be offered an LSI device which encapsulates the complexity involved in being able to converse via the data bus, then the overall result is a simpler system design.

Clearly, when considering the mux bus approach, there must be a commonly accepted set of rules or 'protocols' concerning the nature of the messages, and there is also a requirement for a standard electrical interface, in order to facilitate the provision of standard remote terminal LSI interface devices. The most commonly accepted standard which embraces both these aspects is the US Mil Std 1553B. This has achieved universal acceptance in USA and UK for military aircraft applications and has engendered strong interest outside the airborne field. UK were involved in the formulation of the Mil Std, and it is currently being processed for adoption as a Def Stan (Air).

Mil Std 1553B specifies a serial digital, time division multiplexed, asynchronous, command-response data bus, and we will just examine what is meant by this description. The transmission of data and commands on the bus is serial digital, which means that there is a single bus line - in fact, a balanced twisted shielded pair cable - and all information flowing is bit-serial and word-serial. The interchanges between all subsystems on the bus are interlaced to give a composite pulse train, the appropriate portions of which are recognised and ingested by the intended receiver, and this is what is meant by time division multiplexing.

By referring to the method as asynchronous, we mean that there is no master clock pulse train piped around the system. Each subsystem has its own internal clock, and the messages are encoded on the bus line in what is known as 'Manchester biphase' form. This is a bipolar method whereby encoded 1s and 0s have a transition through zero at the mid point of the waveform, enabling a receiving subsystem to synchronise to its own internal clock. This system asynchrony gives more autonomy to individual subsystems and reduces interconnecting wiring.

By command-response we mean that all transfers on the bus are managed by a bus controller, which commands all traffic and monitors all responses, whether it is actually involved in data exchange or not. Normally, then, the bus control function would be undertaken by a processor, either as a unique task or as part of a larger system task, and the significant point here is that the entire bus data management function becomes a software exercise.

It is not the intention of this paper to go further into the details of mux bus implementation, but we will now consider some of the major advantages which accrue from its adoption in the design of a system. Firstly, a mux system provides a highly ordered vehicle for progressive system integration, particularly at the rig level, as has been demonstrated conclusively by a number of US aircraft projects. The fact that the majority of system traffic flows on a single line means that effective use can be made of a computer simulation rig where actual subsystems are progressively integrated.

Secondly, the data transfer is under software control, validity checking can be undertaken in the intelligent remote terminals at bit, word and message level, and status replies signal the terminals' responses to the controller. All this makes for very high integrity in the transmission and for highly effective error recovery procedures. The system integrity can be further enhanced by the adoption of redundancy, both in the bus line itself and in the bus control area.

Thirdly, the technique has inherent flexibility to accommodate change and growth without excessive hardware modification, because of the simpler interconnection philosophy and the software nature of bus control, and, of course, there are significant reductions in cabling and connector requirements, which makes for even higher integrity. Finally, test and maintenance procedures can be considerably simplified - both built-in test and first line automatic test - because of the ready accessibility of the majority of system data. For built-in test purpose, a system monitor can check on all the bus traffic without actually taking part in the conversations on the bus.

Multiplex data bus techniques have been, or are being, applied in a number of US programs - F15, B1, F16, F18, F11 refit, B52 refit, LAMPS and AAH helicopters and the Space Shuttle. European interest in the technique is evident in the Mirage 2000 implementation and in research programs in UK, Germany and Sweden for future aircraft applications.

5 DISTRIBUTED PROCESSING

In section 2 it was proposed that system complexity could be more manageably contained by distributing processing around the system and, thereby, relieving the large central computing task. There is another aspect of this which arises out of the adoption of mux bus architecture, and that is the need to minimise total system data flow bandwidth.

If the subsystem functions, such as navigation, air data processing, head-up and head-down displays, weapon aiming sensors, communications control, etc, are to be connected by a single multiplex data bus (discounting redundant paths), then it is clear that the bandwidth requirements for the bus could be extremely high, causing significant electro-magnetic compatibility (EMC) problems, both in susceptibility and generation, were it not contained. This is especially relevant when one considers that the actual equipments may be geographically dispersed throughout the aircraft.

Mil Std 1553B specifies a bit rate of 1 Mbit/s, which allows a total capacity on 1 bus of less than 50000 words/s, including command and status words, thereby circumscribing the individual requirements of any subsystem which adopts it. The utilisation of the mux bus, then, demands that data bandwidths are minimised, and this is best achieved by processing raw data at source and transmitting only processed data between subsystems. Clearly, this is another argument in favour of the functional distribution of processing throughout the system, and one criterion in determining the partitioning of that processing.

The central computer concept grew up during the 1960s and early 1970s when the limitations of technology did not make it feasible to have a number of processors in different areas due to constraints of size, weight, power consumption and cost. Micro-processor technology, however, now makes it possible to implement such a system without undue space, weight, power and cost penalties, and it is certain that the architecture of future systems will adopt the distributed processing approach.

At this point, it should be mentioned that the type of distribution envisaged here is that often referred to as 'loosely coupled'. This means that the processing in the various areas is largely autonomous and asynchronous, with data being exchanged via the relatively low bandwidth mux bus. Another type of distribution one could envisage is that known as 'tightly coupled', where the processing elements are concerned with parallel aspects of the same task, are not autonomous, are probably not geographically separated and will probably share common resources such as memory.

This is the typical multiprocessor architecture, and in areas where it may apply in avionic systems it will almost certainly occur at a level below the functional distribution. This means that any one of the autonomous, loosely coupled, functional processing areas could be implemented as a tightly coupled multiprocessing system if the constraints of the task require it. The most likely reason for this is the speed requirement in the individual processing area.

Returning, then, to the loosely coupled avionic system, there are a number of factors which could determine the actual partitioning of the total system processing requirement into the distributed processing areas. Two of these are fairly clear: one factor could be the need to retain functional identity in a particular area, whilst another is the requirement for minimum data flow. Other factors may include the requirement to keep all processing tasks reasonably small, the need to optimise time loading across all areas, the need to retain central system control (known as a 'federated' system), and so on. Optimising any one of these may produce a sub-optimal solution in terms of the others.

What this is saying is that the design of such a system, overall, must be a top-down exercise, because optimisation of the design must proceed in step with the generation of the individual subsystem specifications, and this may well be an iterative process, where the system design is 'tuned'. This is a powerful argument for the adoption of hierarchical requirement statement techniques, and the top-down approach can be carried through into the design of the software to be incorporated into the processing elements.

6 SOFTWARE DEVELOPMENT

Software is clearly becoming a more and more important element of digital avionic systems as the volume of the total processing task increases. Software can represent a significant proportion of system cost, and it can also represent a critical path item in system development, for reasons which will shortly be discussed.

A major improvement in software development techniques has been the widespread adoption of high level language methods for writing programs. There is no doubt that productivity improves and coding errors decrease when such methods are employed. It is, however, not sufficient to specify that a program will be written in Coral 66, say, and hope all problems will disappear. It is still possible to write poor, error-prone software in a high level language as it is to write doggerel verse in the language of Shakespeare. The factors which make software a critical item are poor structure - affecting visibility of design and subsequent maintenance - and inherent errors of logic.

Many surveys have been made into the sources of software error, and it is interesting to note that in large programs nearly two thirds of all errors have been shown to arise from erroneous design rather than straightforward coding mistakes. Even more revealing is the fact that more than 50% of inherent software errors are not likely to be discovered until during or after acceptance testing, and one has to consider why this should be so. One obvious answer is that in the past it has not been considered possible to thoroughly check out software until the target hardware is available, and this is when the software proving becomes a critical path item.

Two important factors to address when considering software development for future systems, then, are techniques to reduce the likelihood of error in software design and coding, and, secondly, methods of advancing the development of software so that it can be tested more thoroughly in advance of the availability of actual target hardware.

Reduction of error must be attacked at the software design level and, here again, a top-down hierarchical approach recommends itself, leading to modularity in design. Many people refer to this as structured programming and what it does is to force a discipline, so that the software design follows basic rules of decomposition, the whole structure is more manageable and visible and, consequently, less error-prone.

The main concept of structured programming is that the program can be designed 'top-down', i.e. it starts as a single statement of the intention of the program as a whole. This level of abstraction is then broken down into a collection of sub-tasks, each of which is wholly independent of the others with all its effects perfectly definable and with no unwanted side-effects. This means that, provided these sub-sections are executed in the correct order, each will perform its own task and will exit at only one wholly predictable point, leaving the data structures ready for the next part, which can only be entered at one point. Each of the sub-tasks is then broken down in a similar manner and the process iterated until a level is reached where no further sensible decomposition is possible. At each level of breakdown, the expression of the tasks performed by each section becomes less and less abstracted and more and more explicit in its nature.

At this stage the conversion of the program to actual source code can usually be very easily performed, making use of such constructs as IF...THEN... or IF...THEN... ELSE... for decisions, and FOR and WHILE loops for iteration. If it is found necessary to break sequence using GOTO statements then the design is not optimum for the 'one way in - one way out' concept, and should be re-examined.

The whole program can, therefore, be expressed in terms of some set of very low level tasks, the effect of each of which is totally predictable; hence, the whole program execution can be predicted and, hopefully, all unwanted effects eradicated. This also leads to a modular design, in which the effects of a modification in one section are easily identifiable within another.

Moving on now to the problem of more exhaustively testing software modules at an earlier stage in total system development, clearly one must make use of host computer facilities together with an executive that manipulates the prototype software in realistic manner. Obviously, the use of a common high level language will facilitate this to a much greater degree than was possible when considering assembly coded programs, but the use of a common executive mechanism is also important, and this may well be provided by the use of the RSRE MASCOT technique.

When considering the total software requirement for a complex system, it can generally be regarded as a number of interdependent but asynchronous processes - air data processing, navigation processing, steering, fixing, etc - and in the final realisation of the system some of these will reside as autonomous tasks in individual processors, whilst others may co-reside in other processors.

Nevertheless, when designing the processing tasks, one can design them as individual modules, making assumptions that the inputs which they require will be satisfied, and defining the types of output they produce. Once all the modules have been so designed, they can be formed into a system by relating specific inputs of some processes with specific outputs of others and by defining their control interactions.

What MASCOT provides is the ability to form the system in this manner and to define the access mechanisms of the data. It can also manipulate the system modules within a host computer as if they were all running asynchronously and simultaneously, by providing the mechanisms for stimulating the processes and accessing the data structures. In other words, it is a real time operating system which enables interacting software modules to be run in the host computer so that many of the inter-relations can be checked, thus enhancing the possibility of tracing design errors at an early stage. This technique, known as prototyping, has been successfully used in a number of software based projects.

Since, at the user level, MASCOT provides a standard real time system approach independent of machine architecture, the final system may well continue the use of MASCOT in processing areas which contain more than one of the asynchronous processing modules, but even if this is not the case, its use during the prototyping phase may make it a valuable tool in the fight against complexity.

METHODOLOGIE DE CONCEPTION D'ARCHITECTURES MULTI-PROCESSEURS POUR DES FONCTIONS D'AVIONIQUE

C. ALEONARD *
A. DEMOMENT
P. ROMAND

J. GILLON **
J.F. LE MAITRE

RESUME

Le but de cette communication est de présenter une méthodologie de conception en technique digitale des systèmes automatiques à hautes performances. En effet, l'évolution des technologies digitales pose aux automaticiens le problème de la conception globale d'un système de commande. C'est-à-dire qu'il faut dépasser le stade de la synthèse algorithmique et prendre en compte, dès le départ de l'étude, tous les aspects de celle-ci : aspect fonctionnel, aspect opérationnel. Ainsi, il est possible d'optimiser les systèmes de commande selon les trois critères importants : le respect des performances fonctionnelles désirées, le coût global et celui, très important, de sûreté de fonctionnement (fiabilité, sécurité, maintenabilité, disponibilité, ...)

INTRODUCTION

Le schéma méthodologique suivi se décompose en deux parties : élaboration de la structure fonctionnelle nominale puis conception, à partir de cette structure nominale, d'une structure redondante remplissant les objectifs de sûreté de fonctionnement. Nous ne présenterons ici que la méthodologie employée pour l'élaboration de la structure nominale fonctionnelle.

MOTIVATION DE L'ETUDE

Les critères de choix, pour une équipe chargée de la définition des moyens de traitement numérique d'un nouvel équipement ou d'un nouveau système, tendent à se modifier rapidement.

D'une part, l'évolution technologique procure une plus grande liberté de choix des constituants et donne moins de préoccupations au niveau de la circuiterie.

D'autre part, les exigences des utilisateurs s'étendent au delà de la simple notion de coût d'achat pour aller vers celle de coût intégré (coût achat, coût exploitation).

Il faut ajouter encore les progrès méthodologiques récents, en automatique particulièrement.

Evolution technologique

L'augmentation de la capacité d'intégration des nouvelles techniques a conduit à intégrer des fonctions logiques très importantes. Il en est résulté deux démarches :

- celle du fabricant de semi-conducteurs qui est contraint d'implanter des fonctions très générales (exemple : microprocesseur, mémoire),
- celle des utilisateurs qui définissent leur propre circuit. Compte tenu de la nature de notre marché cette voie de circuits à la demande est pratiquement interdite.

Nous sommes donc dans la quasi-obligation d'utiliser les produits standards du marché.

Aspect fonctionnel

Un autre aspect concerne la faiblesse relative des performances de ces nouveaux composants. Certes, les progrès de la technologie améliorent de jour en jour ces performances. Mais, parallèlement, les exigences des cahiers de charges vont également en s'amplifiant. Il est donc intéressant d'orienter les travaux nouveaux vers l'étude de structures à plusieurs processeurs. Cette présence de plusieurs organes intelligents, dans un même équipement ou système, amène les avantages suivants :

- il n'est plus nécessaire de faire la course aux produits les plus performants et généralement les plus nouveaux, donc les moins crédibles sur le plan exploitation et souvent les plus coûteux relativement. En effet, le travail simultané (parallèle) de plusieurs processeurs à performances limitées compense la faiblesse des performances individuelles de chaque processeur.
- la défaillance d'un processeur n'est plus critique comme dans le cas d'une structure monoprocesseur. Le fonctionnement coopératif permet d'envisager des possibilités de reconfiguration du système lors d'une panne.

Aspect exploitation

Les coûts d'exploitation sont en forte croissance. Les coûts de la main-d'oeuvre affectée aux interventions de maintenance, les coûts de perte d'exploitation liés à l'indisponibilité du matériel en sont les principaux facteurs.

L'unité "Centrale", à cause de la fonction qu'elle assume, plutôt que par son volume et son coût, est le centre nerveux d'un équipement ou système. C'est donc à son niveau que doivent se situer les efforts en vue d'améliorer les conditions d'utilisation dans les différentes phases de la vie du système :

* Société CROUZET - VALENCE (FRANCE)

** C.E.R.T. - Département d'Etudes et de Recherches en Automatique - TOULOUSE (FRANCE).

- phase étude et développement

La parcellisation du traitement en tâches, l'étude nécessaire de la synchronisation de ces tâches doit permettre un gain appréciable sur le nombre d'erreurs résiduelles en conception, sur la rapidité de développement des logiciels, l'aptitude aux modifications, la certification du matériel et du logiciel.

- phase maintenance

La répartition de ces tâches sur plusieurs unités de traitement offre de nouvelles possibilités en ce qui concerne le test et le diagnostic. Les exigences de sûreté de fonctionnement imposent une détection permanente et "en ligne" des pannes. La reconfiguration nécessite la localisation de cette panne. Ceci montre les améliorations auxquelles on peut s'attendre dans ce domaine.

PLAN DE L'ETUDE

La structure fonctionnelle nominale s'obtient par une approche méthodologique dont les étapes - décrites de manière itérative - sont les suivantes :

- Décomposition fonctionnelle afin d'aboutir à une structure modulaire permettant de simplifier les problèmes de certification du logiciel.
- Choix d'algorithmes orienté vers des solutions minimisant les calculs en tenant compte des possibilités du calcul incrémental, des fonctions tabulées et des problèmes de quantification.
- Graphe de données permettant de représenter les échanges entre modules fonctionnels.
- Graphe de synchronisation tenant compte du temps de réponse et de la cadence d'échantillonnage.
- Ordonnement des calculs et des processus qui est un problème d'optimisation multicritère avec contraintes.

Cette méthodologie est présentée sur un exemple précis d'avionique : celui de la conception d'une fonction anémobarométrique. Cette fonction fut habituellement réalisée en analogique, dans une technologie électromécanique. Plus tard, fut étudiée une version numérique à processeur unique. Cette version à plusieurs processeurs permet d'associer les avantages des techniques analogiques (parallélisme, rapidité, modularité,...) à celle des techniques numériques.

HYPOTHESES

a/ Nous avons volontairement pris le parti d'étudier une structure numérique "collant" à la description fonctionnelle et satisfaisant les exigences opérationnelles (aspect automatique) plutôt que considérer la réalisation d'une structure universelle sûre de fonctionnement (aspect informatique). Nous pensons retrouver dans de telles structures les avantages de l'analogique (chaînes fonctionnelles, etc...) sans perdre les avantages du numérique (pas de dérives ...) et sans avoir les inconvénients du numérique (le processeur ou l'ensemble de traitement qui est un point de passage forcé pour toutes les informations). En d'autres termes il est préférable de faire en sorte que ce soit l'objectif (l'application) et non le moyen (le processeur) qui détermine la structure du système.

Cette approche permet d'attendre de nouveaux avantages :

- en certification des logiciels : les modules sont courts, avec peu d'entrées/sorties et faiblement synchronisés,
- en coût : le système est taillé "sur mesure" pour l'application et ne supporte pas la lourdeur des systèmes universels avec leurs problèmes de gestion interne etc...

b/ Nous avons fait a priori le choix technologique d'utiliser les microordinateurs monolithiques qui intègrent sur la même "puce" : processeur, mémoire, entrées/sorties (cf. INTEL 8748, MOSTEK 3870, MOTOROLA 6801, etc...). Ce choix a priori peut se justifier par les gains en volume, coût, consommation, fiabilité amenés par ce type de composants.

Les microordinateurs ont une mémoire programme qui est morte. Ceci n'est pas une contrainte dans notre thème de travail car l'allocation des tâches est figée.

Nous avons également préféré l'emploi de microordinateurs identiques pour toutes les tâches, plutôt que celui d'un élément technologique adapté à la tâche à traiter (exemple : emploi du circuit de calcul AMD 9511 qui réalise les opérations de multiplication et division), ceci pour les raisons suivantes :

- l'emploi de circuits spécialisés aurait considérablement accru les taux d'échanges entre processeurs. En effet, le préalable à toute tâche aurait été de transférer dans ce processeur spécialisé les données afférentes à celle-ci,
- l'emploi de composants processeurs identiques facilite les reconfigurations. Leur banalisation permet l'exécution d'une tâche, en cas de panne, sur n'importe quel autre processeur (à des problèmes d'accès aux informations près),
- nous pensons également tirer un gain, sur le plan approvisionnement par ce choix de processeurs identiques.

c/ Nous avons retenu, et dans l'ordre, les critères de choix suivants pour sélectionner une solution

- contraintes fonctionnelles,
- contraintes de sûreté de fonctionnement,
- facilité de mise au point et maintenance,
- coût,
- volume, consommation,
- communications entre processeurs les plus faibles possible,
- équirépartition des tâches sur chaque processeur.

APPROCHE METHODOLOGIQUE

Décomposition fonctionnelle

A une fonction définie dans le cahier des charges est associée une tâche ou un ensemble de tâches. Cette décomposition est faite indépendamment de tout choix de réalisation. Elle représente l'analyse du problème posé et elle a pour but de définir la modularité d'un problème en vue d'une meilleure compréhension de celui-ci et d'une meilleure résolution (certification,) (cf. Figure n°1).

Choix d'algorithmes/Choix technologique

- Ce choix n'est pas indépendant d'un choix technologique. Le choix technologique peut précéder la détermination des algorithmes. Ce choix a priori nous semble plus proche de la réalité car souvent le processeur est imposé par une liste préférentielle. Seul un échec peut conduire à remettre en cause ce choix initial. (démarche itérative).

- Afin d'obtenir :

- . une réduction de coût,
- . un volume matériel convenable,

le choix s'est porté sur l'emploi de macrocomposants standards. En particulier, les unités de traitement choisies sont des microordinateurs (ou microcalculateurs) monolithiques (type INTEL 8748, MOSTEK 3870,...) qui intègrent dans un même boîtier le processeur, la mémoire programme, la mémoire données et les entrées/sorties. Ces microordinateurs ont un format de mot de 8 bits. Ils sont relativement lents mais sont peu coûteux. Le parallélisme dans l'exécution des tâches compense les faibles performances individuelles de chaque microcalculateur (nous les appelons μ c dans la suite du texte).

Le 8748 d'INTEL a été retenu car c'est le seul composant microcalculateur actuel qui, dans le format 8 bits, ait à la fois des spécifications précises et des possibilités d'extension mémoire et entrées/sorties. Le MOSTEK 3870 ne tolère pas d'extension mémoire externe. Le MOTOROLA 6801 n'a pas de spécifications précises actuellement, tout au moins à notre connaissance.

- Les méthodes de calcul doivent être adaptées à ce type de matériel. Le but de cette étape est de déterminer les besoins en nombre d'instructions et en volume de données. Les méthodes de calcul envisageables sont les suivantes : calcul incrémental, calcul à base d'interpolation sur des tables ...

Les calculs doivent s'inscrire dans des contraintes temporelles. Les normes spécifient souvent les grandeurs de sortie sous forme analogique en définissant l'ordre de la fonction de transfert, la fréquence de coupure f_c . Il est donc nécessaire de transformer ces spécifications analogiques en spécifications numériques : un temps de réponse " δ " qui définit l'intervalle de temps entre un échantillon d'entrée et l'échantillon de sortie correspondant, un taux d'échantillonnage " τ " qui définit l'intervalle de temps entre deux échantillons d'entrée (ou de sortie).

Les deux inconnues τ et δ sont calculées en écrivant les contraintes de phase et d'amplitude :

$$\delta = \tau < \frac{K}{f_c} - K' \quad \text{avec} \quad \begin{array}{l} K = \text{constante} \\ K' = \text{constante} \end{array}$$

δ et τ sont considérés égaux pour faciliter la mise en oeuvre de la loi de commande du système bouclé. (cf. Figure N°2).

GRAPHE DE DONNEES

A chaque loi ou variable, il est associé une tâche. Le graphe de données représente les échanges entre tâches fonctionnelles. L'examen du graphe de données peut mettre en évidence certaines possibilités de redondance algorithmique, c'est-à-dire l'élaboration d'une fonction sous deux formes différentes. (cf. Figure N°3)

GRAPHE DE SYNCHRONISATION (résultant du graphe de données)

Un choix est fait sur le mode d'échange des données entre tâches. Ce choix va imposer certaines contraintes sur la synchronisation entre tâches. Par exemple, si l'on suppose que les tâches communiquent par un mot mémoire, ce mot mémoire est partagé par deux tâches connexes. La tâche amont accède à ce mot mémoire en écriture. La tâche aval y accède en lecture. Ce choix va donc créer des conflits potentiels d'accès à ces mots mémoire. Les tâches sont donc considérées comme des opérateurs élémentaires qui rendent indisponibles leurs entrées/sorties pour toutes les tâches en conflit sur ces mémoires pendant toute la durée de leur exécution. Une tâche est alors exécutable si et seulement si toutes ses entrées/sorties sont disponibles. Lors de l'exécution d'une tâche, les entrées/sorties de celle-ci sont rendues indisponibles. A partir du graphe de données et des contraintes du spécificateur on tire un graphe de synchronisation à parallélisme maximal. (La représentation retenue est un RDP, le fonctionnement est du type Pipe-Line). (cf. Figure N°4).

L'évaluation de chaque tâche en temps d'exécution et en bilan mémoire est nécessaire pour faire un choix définitif de synchronisation. Dans le cas de la fonction anémobarométrique, une synchronisation type monocadencé ne permet pas de répondre aux exigences temps réel. Il a donc été retenu un mode multicadencé qui satisfasse les contraintes temps réel et qui ne complique pas trop la gestion des tâches. Ceci aboutit à la définition de sous-graphes monocadencés : un sous-graphe par cadence et du nombre de processeurs nécessaires à la résolution du problème.

ORDONNANCEMENT

Les données du problème sont les suivantes :

- le graphe de synchronisation
- le temps d'exécution des tâches
- les cadencements à respecter

- d'autres contraintes : par exemple, le volume mémoire adressable par le microcalculateur.
- des critères à optimiser.

Le problème est de répartir ces différentes données sur les différents processeurs et les ordonner dans le temps (répartition spatiale et temporelle).

Il n'existe pas à l'heure actuelle de méthodes simples pour un examen exhaustif des différents ordonnancements possibles. On s'oriente donc vers la recherche d'ordonnancements sous-optimaux à l'aide d'heuristiques. Cette démarche donne la plupart du temps une solution valide acceptable quand ce n'est pas optimale. (cf. Figure N°5).

CONCLUSION

Cette méthodologie aboutit à la définition d'une architecture fonctionnelle nominale à partir de laquelle peuvent être étudiés les impératifs de sûreté de fonctionnement. Celle-ci est indispensable pour l'étude des structures à plusieurs processeurs et même pour les structures monoprocesseurs. Le problème à résoudre est mieux segmenté, mieux analysé et en particulier, lors de modifications, l'impact de celles-ci peut être mieux évalué.

D'autre part, la recherche de structures numériques ajustées au fonctionnel conserve les avantages de l'analogique sans perdre les avantages du numérique. Enfin, ces structures permettent d'attendre de nouveaux avantages en certification des logiciels et en coût.

SORTIE	BANDE PASSANTE à 3 db (2e ordre) en Hz	PROBABILITE DE PERTE DE LA FONCTION = FIABILITE SIGNAL	COMMENTAIRE
Vc	3	proba $< 10^{-7}/h$ vol	- Panne dange- reuse
Qc	3		- Perte extrê- mement rare
Zp	3	$10^{-7}/h$ vol $<$ proba $< 10^{-5}/h$ vol	- Panne majeure:
M	3		
Ps	3		- Perte rare
α_v	5		
Vz	0,5	$10^{-5}/h$ vol $<$ proba	- Panne mineure
VMO/MMO	0,5		
Zc	3		- Perte probable
Tt	0,1		
Ts	0,1		
Vp	0,5		
α	0,5		
QFE/QNH	Statique		

FIGURE 1 : Contrainte de bande passante et de fiabilité.

SORTIES	f_c Hz	CONTRAINTES $\partial = \tau$ ms	CHOIX $\partial = \tau$ ms
α_v	5	33	39
Ps Qc M Zp Zc Vc	3	39	39
Vz VMO/MMO Vp σ	0.5	316	1248 312
Tt Ts	0.1	1650	312

FIGURE 2 : Détermination des temps de réponse et période d'échantillonnage

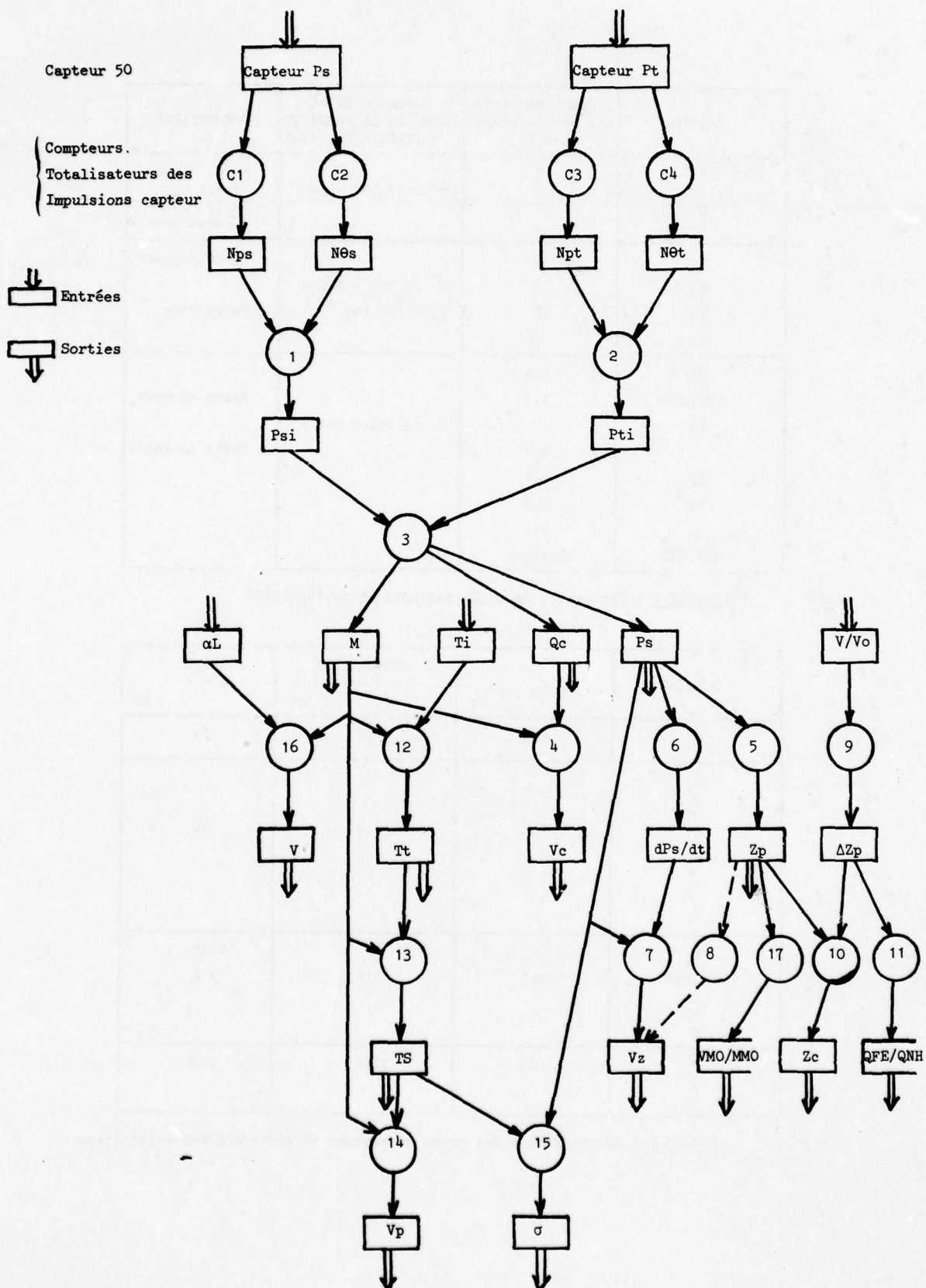


FIGURE 3 : Graphe de données de la fonction centrale

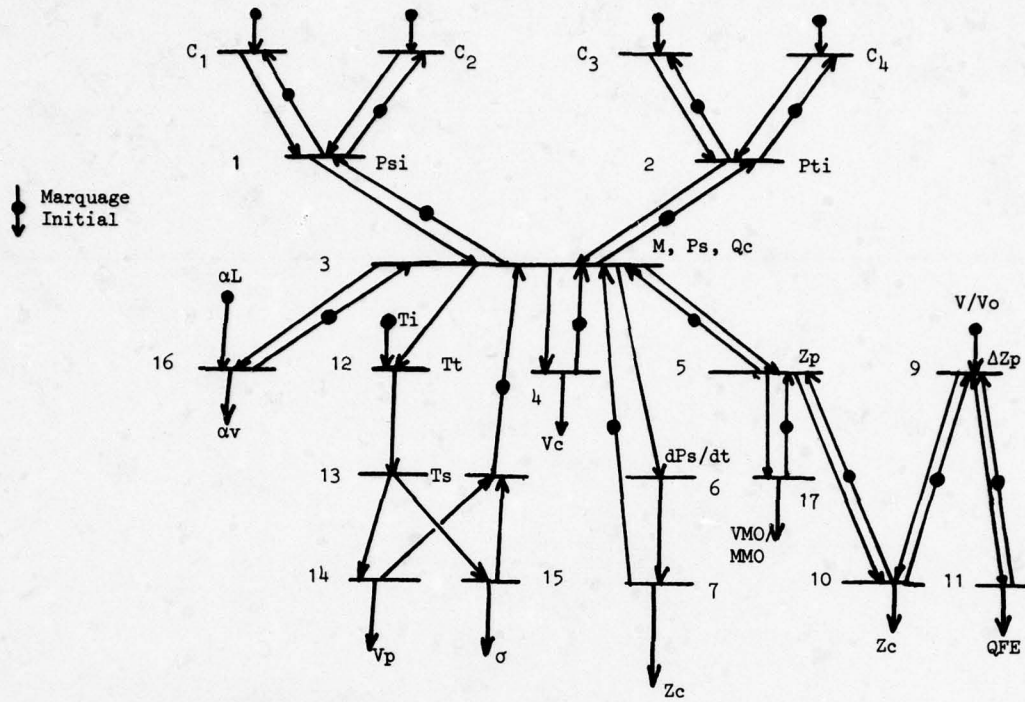


FIGURE 4 : Graphe de synchronisation de la Centrale

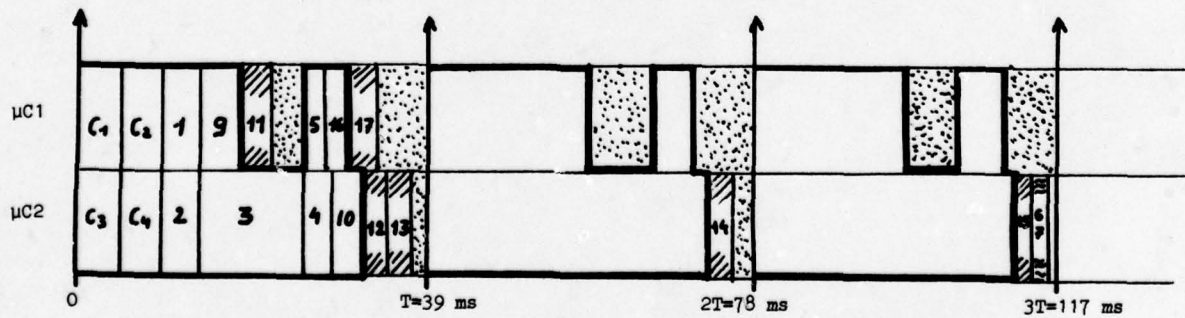


FIGURE 5 : Ordonnancement retenu sur les deux microprocesseurs (optimisation des échanges).

FORTRAN FOR AVIONICS

Austin J. Maher
The SINGER Company
Kearfott Division
Wayne, New Jersey
USA

ABSTRACT

The use of High Order Languages (HOL) for avionics applications has taken a long time to gather momentum. At last it appears to be an idea whose time has come and should have come much sooner. It has finally arrived for a number of reasons but a prime causal agent is certainly the recent DOD Directives which mandate the use of an HOL and limit the languages considered to seven (FORTRAN, COBOL, J3, J73, CMS2, TACPOL, SPL-1). Probably equally important in this regard are the technological advances in avionic computer architecture and LSI components which make the price paid for HOL use in this environment acceptably small. Whatever the reasons, it now behoves every supplier of avionic computers and every supplier of avionic systems with embedded computers to develop an aggressive technical approach to the use of HOL for avionics computation.

But what language(s) should be considered? The efforts by ARPA to develop a Common DOD High Order Language promise to eventually resolve this question with a single, comprehensive, modern HOL which includes providing contractors with the ability to develop a code generator for the new language at a very low cost (approximately \$25,000). Achievement of this goal will be a major factor in the widespread acceptance of the new language. But the lead time for this new language is appreciable. So contractors are faced with the question, What approach should we adopt for HOL use in the interim? Since developing compilers for all six interim languages (omitting COBOL) is too expensive, what interim language(s) should be addressed?

This paper describes the approach taken by one contractor to address this problem. It includes a review of the interim languages stipulated by the DOD and the reasons for selecting FORTRAN as the basis of a company-funded compiler activity to provide competitive HOL capability during this interim period. The presentation includes the rationale for selecting the oldest and least modern of the six interim languages and the design approach to select a suitable language dialect for efficient implementation of avionic software without violating the stipulations of the ANSI Standard for FORTRAN. Finally the exposition covers the success of the effort to develop a low-cost compiler for the FORTRAN dialect which generates highly efficient object code in the tradition of the many highly efficient FORTRAN compilers which have been developed for commercial computers.

BACKGROUND

Singer-Kearfott began investigating the use of High Order Languages (HOL) for avionic computer programming in 1964 when a NELIAC compiler was developed for one of our early avionic computers. This early avionics compiler did not meet with widespread acceptance due to the inefficiency of its object code, a common fate for contemporary avionic compilers. One conclusion of this activity was the realization that the addition of expensive optimization algorithms to the compiler would NOT solve the inefficiency problem since much of the inefficiency could be traced to the architecture (instruction set) of the avionics computer. In 1969 the development of the SKC2000 computer was begun. This computer was designed to eliminate the inefficiency caused in HOL object code by the architecture of the computer. The singular success of this design was proven when the SKC2000 (AN/AYK13) was used for the central GN&C functions on the B-1 aircraft using the JOVIAL J3B language. Based on this experience, Boeing concluded that the SKC2000 computer was 25% more efficient than its nearest contemporary competitor in both speed and memory utilization.

Based on this successful experience, among others, the U.S. Department of Defense decided to encourage the use of HOL for avionic computer software and other computers embedded in weapon systems. Toward that end, DOD Directive 5000.29 (reference 8) was issued in 1976, stipulating that an HOL must be used in the development of weapon system embedded computer software unless a life cycle cost analysis revealed that it was not cost effective. Shortly thereafter, the DOD issued Instruction 5000.31 (reference 9) which addressed the uncontrolled proliferation of languages used in weapon system applications. Until that time, a wide variety of languages had been used with little tendency to standardize. This situation discouraged weapon system computer suppliers from investing in the development of a compiler for their products since the probability of realizing a return on their investment was slight.

By 1976 Singer-Kearfott had decided that it was feasible and cost effective to use an HOL for the development of software for the avionic microcomputers (the SKC3000 series) used in their GN&C products, including products for: inertial navigation, doppler navigation, TDMA communication, weapon delivery, ICNI, hybrid navigation, missile guidance, and flight control. But first it was necessary to apply the lessons learned with the SKC2000 computer in the generation of efficient object code to these subsystem applications. Toward that end, the design of the SKC3120 microcomputer, which used commercial bit-slice microprocessors, was adapted to handle an enhanced version of the SKC2000 (AN/AYK13) instruction set and was redesignated the SKC3121 one-card microcomputer. Through the dramatic advances in LSI circuit technology, it was now possible to implement a superset of the proven SKC2000 instruction set on a single electronics card.

Since it was decided to use this enhanced instruction set for all Kearfott computer and microcomputer products, it was desirable and cost-effective to develop an HOL compiler for use in this wide range of subsystem products. Such a development would permit Kearfott to reap the benefits of an HOL in all subsequent system designs, including those for which an HOL would not be cost-effective if it were necessary to fund the development of a compiler for the candidate system alone.

LANGUAGE SELECTION

But what language should this company-funded compiler process? Since the DOD Instruction 5000.31 listed only seven languages which would be acceptable for future weapon system embedded computer software, we constrained our selection to one of these seven languages. One of them, COBOL, was immediately eliminated since it was clearly inapplicable to the computations required of Kearfott's GN&C products, having been developed for business data processing. There remained six acceptable languages to evaluate, namely:

- 1) JOVIAL J3 - developed by the US Air Force for Command & Control applications
- 2) JOVIAL J73 - developed by the US Air Force for avionics applications
- 3) CMS2 - developed by the US Navy for shipboard Command & Control
- 4) SPL/I - developed by the US Navy for signal processing
- 5) TACPOL - developed by the US Army for gunfire control
- 6) FORTRAN - the most widely used HOL for scientific applications

We reluctantly omitted the J3B dialect of JOVIAL from consideration because the DOD did not include it in the list of acceptable languages despite its record of significant success in the B-1 application (with the SKC2000 computer) and the F-16 application (with the Delco M362 computer) prior to the issuance of DODI 5000.31 and its selection for the B52G update on a waiver basis. More recently (1978) we have seen the USAF develop a revised definition of J73 to include the best features of the J3B and J73/I dialects of JOVIAL to yield a superior language for avionics applications. Their plans also include the development of software to support a largely automatic translation from J3B or J73/I to the syntax of the revised J73 language. If carried to fruition, this effort promises to provide a single JOVIAL language for Air Force avionics applications and which may (perhaps with some further modification) also supplant the J3 dialect. However, these developments were not known in 1976 when the language selection for Kearfott's HOL compiler was being made, so while J3B was included in our tabulations of language features it was not seriously considered due to its rejection at that time by the DOD.

After considering all the factors which were deemed relevant, the FORTRAN language was selected for implementation. This was a somewhat surprising choice since all the Kearfott software engineers consulted in the study were initially reluctant to seriously consider FORTRAN due to its image of being an archaic language with no "realtime" features and which is singularly unsuitable for structured programming. However, these apparent disadvantages were subsequently successfully addressed after FORTRAN became the leading candidate based on the considerations outlined in Table I and discussed below:

- 1) ANSI Standard FORTRAN has been successfully used since 1967 by Kearfott for realtime programs used in automatic test equipment. This experience led to the conclusion that the language is acceptable for avionics work. However, all the other candidate languages were considered superior to FORTRAN for avionics applications with the languages J73 and SPL/I being particularly well suited since they both were modern languages with the appropriate structured programming constructs as well as the arithmetic and logical constructs which are necessary for programming control systems.
- 2) FORTRAN has been almost exclusively used by Kearfott for avionic system performance simulation and algorithm development. The use of FORTRAN for programming the avionic computer would permit many airborne computer

2) Continued

- source modules to be directly used in these simulations thus achieving a very significant cost reduction. This cost reduction would result not only from the savings in the cost of programming the module twice, but also from the decrease in program bugs which must be detected in the expensive lab integration activity due to the fact that the simulated module is not identical to the operational module. Further cost reductions accrue because Kearfott's System Analysts are already fluent in FORTRAN and are therefore well equipped to participate in walk-throughs of the operational program to detect conceptual errors at an early development stage. Furthermore, the simulation advantages envisioned by SKD would likely accrue to our customers as well, since their simulation programs are probably also in FORTRAN and their personnel well versed in its use.
- 3) The cost of developing a compiler is a very significant factor when the funding source is a limited IR&D budget, so the relative cost of a compiler for the various languages was given serious consideration. We were particularly interested in developing a compiler which would be host computer portable since we intended to deliver the product to a variety of unspecified customers, with varying host computer types and configuration. The cost of developing a portable FORTRAN compiler was determined to be significantly less than the cost incurred in the development of any other compiler. For example, the cost of developing the J3B compiler for the SKC2000 was approximately \$400K. The cost of compilers for the other languages would be even higher, in some cases approaching \$700K. FORTRAN compilers are generally much less expensive, since so many of them have been developed that their development techniques are well known and in many cases a specific design exists. Therefore Kearfott was concluded to have sufficient resources for independent development of a FORTRAN compiler. J3B is the only language whose compiler development costs approach that of FORTRAN. As mentioned earlier, an excellent J3B compiler has already been developed by Softech for Kearfott's SKC2000 computer. However, the compiler is only portable to host machines with a compatible AED compiler since the J3B compiler was developed in AED language. Development of a truly portable version would be more expensive than a FORTRAN compiler.
 - 4) It was Kearfott's plan to develop the selected compiler over a two or three year period with incremental funding from each year's IR&D budget. If the design were conducted at Kearfott facilities, not only would incremental funding be facilitated but it would encourage effective joint optimization of the compiler code generator and the airborne computer's instruction set. Consequently, we determined that another significant factor in the selection was the relative suitability of the candidate compilers for development at Kearfott facilities with incremental funding. FORTRAN was particularly well suited for development in this fashion since its design could be easily handled by personnel who had worked on earlier support software for Singer-Kearfott airborne computers. Even J3B required a substantial subcontract to adapt its compiler to the SKC3121 instruction set and to make it host machine portable. All other candidate languages were even less suitable in this regard since they did not have the benefit of an existing SKC2000 compiler to serve as a baseline.
 - 5) Training costs would be practically non-existent for both Kearfott applications programmers and customer personnel. Even new graduate engineers now have a significant knowledge of FORTRAN so it would be a simple matter to become acquainted with the subset dialect to be selected by Kearfott.
 - 6) FORTRAN is the only language known to be in use by all the DOD services and NASA. Consequently, an investment in FORTRAN is not uniquely directed at a language used only by one agency. While there is no assurance that the FORTRAN language is acceptable for any particular future procurement, its use will likely receive serious consideration if a FORTRAN compiler were readily available from SKD (or any other computer or system vendor). The investment in any other single language would be much less likely to be accepted since agencies other than the original language sponsor would probably not consider its use.

Now that we have identified these significant advantages for the selection of FORTRAN as SKD's avionics HOL, what about the negative aspects mentioned earlier? Before selecting the FORTRAN language we carefully considered each of these "problems" and resolved them as outlined below:

- ° Lack of "realtime" facilities - While FORTRAN is lacking in special constructs to enhance its suitability for realtime programming, this deficit has not proven to be a significant deterrent to Kearfott's widespread use of FORTRAN for realtime programming in the minicomputer environment (e.g., HP2100, PDP-11, etc.). Since 1967 we have made extensive use of FORTRAN to develop realtime programs for automatic test equipment and special purpose data reduction laboratory facilities. Whatever disadvantages are presented by the FORTRAN language in this environment have been easily circumvented by the occasional use of Assembler language modules. In no case did the magnitude of this assembler language code become so appreciable as to warrant reconsideration of the use of FORTRAN for the majority of the code. So it would seem that the FORTRAN language is more than adequate for realtime applications. In fact it is possible to augment the basic capabilities of FORTRAN with special realtime features (as will be discussed later) without sacrificing ANSI Standard compatibility.
- ° Unsuitable for Structured Programming - FORTRAN, as defined by the 1966 ANSI Standard (reference 3), was not a block structured language and therefore did not include the basic Structured Programming constructs (IF THEN ELSE, DO WHILE, etc.) but did include the infamous GOTO statement which permits (encourages?) development of unstructured code. Our first reaction to this allegation was the realization that the absence of the block structured constructs did not preclude the design of well structured FORTRAN programs. In fact, there were many examples of properly structured FORTRAN programs in our recent experience. Our second reaction was the realization that the on-going FORTRAN standardization activity had recognized this problem and was taking steps to alleviate it. Since then, a new ANSI Standard has been issued (references 2, 4, 5) which defines FORTRAN 77 to include an IF THEN ELSE form, as well as other language improvements. The standards committee is also continuing to work on further language improvements (including DO WHILE, CASE, local Procedures, etc.) for inclusion in the next version of the standard, expected to be released in about 1982. It is clear, then, that the FORTRAN language will continue its evolution toward the more modern languages such that their differences will gradually fade away as will the rationale for rejecting the selection of FORTRAN on this basis.

Having thus resolved the two negative allegations against FORTRAN, we were able to select it as the baseline for the language to be processed by the Singer-Kearfott HOL avionic compiler. We further decided to select a subset of the language for implementation which was particularly well suited for avionics applications but which minimized the complexity of the compiler. The compiler was designed to facilitate inclusion of the new language constructs with minimum effort. However, to preclude the development of a non-standard dialect, the new constructs should not be included until they have been incorporated in a released ANSI Standard or at least until the standards committee has stabilized on the recommended construct.

DESIGN OF THE FORTRAN LANGUAGE

Once the selection of FORTRAN as the baseline HOL for avionics was made in 1976, it was next necessary to tune the standard language somewhat to optimize it for avionics applications. Our goal in this tuning phase was to develop a pure subset of ANSI FORTRAN which would hopefully also be a pure subset of the yet to be released FORTRAN 77 ANSI Standard. Where it was desired to add a capability which is not included in the ANSI Standards, the intention was to add it in such a fashion that compatibility with the ANSI Standards was preserved. This approach should circumvent one problem encountered in the 1966 ANSI Standard where the smaller version of FORTRAN defined by the standard and designated Basic FORTRAN was NOT a pure subset of the full FORTRAN language which it also defined. These goals were very well met by the realtime avionics dialect of FORTRAN which was developed for Singer-Kearfott computers and which was designated SKC FORTRAN.

The first problem addressed in defining SKC FORTRAN was the existence of pathological syntax problems in ANSI FORTRAN which date back to its original syntax definition in 1957, many years before our ability to analyze language structure had matured. Based on theoretical developments since then we now see that the majority of these pathological syntax problems in FORTRAN can be traced to a single FORTRAN syntax feature, its treatment of the "blank" character. The original definition of FORTRAN permitted a blank character to be inserted or deleted from a source program in any position (except Hollerith strings) with no effect on the interpretation of the source code. In other words, the syntax definition was to be entirely insensitive to the occurrence of blank characters in the character string which defines the FORTRAN source program. Since this feature of the FORTRAN grammar served no apparent beneficial purpose and since it significantly complicated the compiler design task, we decided to make a

modification for SKC FORTRAN which solves the pathological syntax problem while retaining SKC FORTRAN as a pure subset of the ANSI Standards. It was decided that the blank character should have some limited syntactic significance. In SKC FORTRAN it is not permitted to embed a "blank" within a keyword or variable name. Also, the first keyword in a statement must be terminated by either a blank or a FORTRAN punctuation character (such as + - * /). These simple restrictions, which correspond to almost everyone's normal coding practice, served to eliminate the difficult syntax problems of FORTRAN, while permitting SKC FORTRAN source code to be compiled by any ANSI Standard FORTRAN compiler without difficulty (the proof that pure subsetting had been retained).

The next issue addressed in designing the SKC FORTRAN language was input/output. How should the language support the I/O needs of avionics applications? An important factor here is the fact that input/output requirements for avionic applications can vary over a very wide range. For example, I/O for an avionics application may be limited to the rudimentary binary I/O typical of small unmanned vehicles or may consist of a complex I/O system with mass memory communication and heavy human interface typified by the B-1 central GN&C system or may consist of multiple bus interfaces to achieve highly reliable, fault-tolerant I/O typified by the Space Shuttle GN&C system or a digital fly-by-wire system for aircraft. Due to this wide disparity in I/O requirements, it was decided to delete the conventional FORTRAN input/output constructs from SKC FORTRAN, leaving that function to be handled by subprograms which could be easily called by SKC FORTRAN and which could be tailored to each application. In this regard we followed the precedent set by all JOVIAL compilers and most realtime languages and retained our adherence to strict subsetting with ANSI Standard FORTRAN since all I/O operations in SKC FORTRAN appear as normal FORTRAN subprogram calls.

ANSI Standard FORTRAN also contains a couple of features which are well known violations of good programming practice, as well as being difficult to implement. The first of these is the capability to leave the range of a DO loop using a GOTO statement and then to return to the range of the loop while retaining all loop counters intact. This capability is referred to as the "Extended Range of a DO loop" and can be the cause of singularly unreadable code. The second such feature is the full FORTRAN EQUIVALENCE capability, which permits variable names to be equated thus causing a sharing of a memory location by the two variable names. When the EQUIVALENCE is used for entries in tables or arrays, it is very possible for memory allocation problems to arise since the equating of names then reduces to an implicit decision by the programmer to allocate memory manually. In many cases, the HOL programmer is not fully aware of all the constraints which must be satisfied by a correct memory allocation, nor should he be. This can lead as a minimum to FORTRAN code whose intent is not easily discerned or to obscure programming errors in the worst case since the memory allocation actions are difficult for the HOL programmer to predict. Since each of these features is a prime source of poor programming practices, is difficult to implement, and provides only slight redeeming benefit, it was decided to delete both from the definition of SKC FORTRAN.

We next considered two features of Standard FORTRAN which have applicability to some avionics applications but which are implemented in such a special fashion in FORTRAN that we were hesitant to invest money to include them in SKC FORTRAN. The first of these is the very limited Hollerith data capability found in FORTRAN. Since we knew that the ANSI Standards committee was planning to incorporate the CHARACTER data type in FORTRAN 77 and since this is a much more satisfactory approach than the limited Hollerith constructs, we decided not to invest in Hollerith constructs for SKC FORTRAN. A similar situation existed for the Statement Function in FORTRAN. The Statement Function provides the capability to define a callable Function using a single FORTRAN statement. Since the Statement Function can be significantly more efficient than the more conventional Function Subprogram (due to implicit argument transmission), it represents an attractive capability for avionic programming. However, since its scope is limited to a single assignment statement, its applicability is relatively infrequent. After much consideration it was decided that the Statement Function would not be implemented in SKC FORTRAN due to its restrictive definition. However, we also decided to define a more general approach to the implementation of local subprograms in FORTRAN and to implement this capability in a subsequent version of SKC FORTRAN. This definition was prepared under the title Local Procedures and was incorporated in the Language Reference Manual (reference 1). A copy of the definition was supplied to the ANSI Standards committee, on their request, for consideration in the next update of ANSI Standard FORTRAN. We believe the Local Procedures capability provides a more powerful alternative to the Statement Function capability.

In addition to the subset restrictions described above, we also decided to eliminate or delimit a few FORTRAN language features due to their inapplicability in the typical avionics problem. Among these are the decisions to delete the COMPLEX data type and to limit arrays to two dimensions initially and three in the later version of SKC FORTRAN.

Finally, we addressed the various proposals to extend the SKC FORTRAN language to provide new capabilities, including the ability to handle packed single bit variables, angles expressed in Binary Angular Measure (BAM), logical operations and the handling of large data tables in memory. These issues were approached very conservatively due to our strong desire to retain compatibility with the ANSI Standard. The principal conclusion of these deliberations was a compromise approach which permitted all the listed capabilities to be provided without violating the ANSI Standard. The desired capabilities were translated into the form of a set of intrinsic functions which would be recognized by the SKC FORTRAN compiler, which would in turn generate tailored in-line code for each function, often based upon a special SKC3121 instruction designed for this purpose. The source code which invokes these special functions could also be compiled by any ANSI FORTRAN compiler and executed in conjunction with a run-time support library to which these FUNCTION Subprograms have been added. Since the added FUNCTION's can be easily written (even in ANSI FORTRAN if desired), we have succeeded in maintaining strict ANSI compatibility although the capability of the SKC FORTRAN compiler has been significantly enhanced for avionic problems while retaining very high memory and speed efficiency. It is interesting to note that the Purdue Committee for Industrial Realtime FORTRAN (reference 6) adopted this identical approach for most of these "bit manipulation" functions, subsequent to Kearfott's selection of this approach in 1976. Consequently, SKC FORTRAN is highly compatible with the proposed definition of Industrial Realtime FORTRAN.

A couple of proposed extensions were not implementable as FORTRAN FUNCTION's. Most modern FORTRAN compilers permit mixed mode arithmetic expressions and it was desired to include this feature in SKC FORTRAN although it was not permitted in the 1966 ANSI Standard. The 1966 Standard also limited the length of the names of FORTRAN variables to six characters. This presented a significant restriction to the use of readable variable names which is not present in any modern programming language. It was decided to incorporate both of these features in SKC FORTRAN despite the fact that they were not covered in the 1966 Standard since it was highly likely that both would be included in the FORTRAN 77 Standard and both would be very difficult to implement in SKC FORTRAN on a retrofit basis. Even if they were not included in the new Standard, they did not clash with any competitive language forms so no duality of meaning would be encountered in the future (as was encountered with Basic FORTRAN in 1966) and it would be an easy matter for SKC FORTRAN programmers to avoid these extensions by convention if strict ANSI FORTRAN code was desired. In the final analysis we were 50% successful in this gamble, since mixed mode arithmetic WAS included in the FORTRAN 77 Standard but it did not permit variable names longer than six characters. We hope this singular omission will be corrected in the next update to the FORTRAN Standard.

One other feature of the selected SKC FORTRAN language semantics is different from many commercial FORTRAN compilers although it is not a violation of the 1966 ANSI Standard (which is silent on the subject). The indicated feature is the storage of local subprogram variables in a temporary data area which is allocated on entrance to the subprogram and released upon exit from the subprogram. This feature is indispensable for the development of reentrant subprograms which are of great utility in realtime computer applications. Ordinarily this feature is transparent to the programmer, unless he assumes in his program that his prior local variable values have remained unchanged between two invocations of their defining subprogram. So long as he avoids such an assumption in his program, no problems shall be encountered in using the same subprograms in both SKC FORTRAN and commercial FORTRAN compilers. It is important to note that the ANSI Standard for FORTRAN 77 has corrected the omission in the 1966 Standard and has stipulated that local variables in subprograms be allocated in temporary memory, and so is compatible with SKC FORTRAN.

In summary, then, we have designed an avionics dialect of the FORTRAN language which is a strict subset of FORTRAN 77 except for the inclusion of long variable names and Local Procedures. The first of these features has been implemented in Version 1 of the compiler and it is tentatively planned to implement the second in Version 2 of the compiler. We also believe there is a high probability that the next update to the FORTRAN Standard will remove these deficiencies.

RESULTS AND CONCLUSIONS

The development of Version 1 of the SKC FORTRAN compiler was successfully concluded in late 1978. It is currently undergoing its initial shakedown on a couple of in-house activities prior to its use on several proposed system developments for external customers. To date we have seen a clear demonstration that the major design goals were effectively met and have been particularly gratified by the high level of object code efficiency achieved without global optimization and prior to the incorporation of the register allocation directives. To illustrate this point, we are conducting a series of benchmark analyses to quantify the code efficiency of the compiler. A partial summary of these results is shown in Table II. This tabulation compares the size of the object code required for three benchmark routines which are encountered in avionics systems programming. The computers covered include the IBM 370, the SKC3121 and several commercial minicomputers. Note that three of the compilers used were the result of very substantial investments to achieve automatic optimization, namely: FORTRAN H Extended for the IBM 370, FORTRAN Plus for the PDP-11 and FORTRAN VII for the Perkin-Elmer (Interdata) computer. However, in all cases they were beaten by the SKC FORTRAN compiler,

often by substantial margins. While it is obvious from these figures that the SKC FORTRAN compiler does not introduce any significant inefficiency in the object code, we do NOT mean to imply that the SKC FORTRAN compiler does a better job of optimization than its expensive competitors listed above. The primary reason for the superior object code statistics is the architecture (instruction set) of the Singer-Kearfott computer products. In fact, if a comparable investment in global optimization were made for the SKC FORTRAN compiler, we would expect these benchmark results to improve even further.

With these goals clearly accomplished, it remains now to demonstrate that the goals of host computer portability and HOL flexibility were also successfully met. Since we have high confidence in both areas, we are considering two activities which will clearly verify the compiler's performance in these areas. The first is the transporting of all Kearfott computer support software, including the SKC FORTRAN compiler, to a large minicomputer host processor, the PDP-11. The second is the adaptation of the compiler to handle a subset of the revised JOVIAL J73 language, once its specification is finalized. These two efforts, if successfully concluded, will clearly demonstrate the accomplishment of every compiler design goal.

REFERENCES

- 1) SKC FORTRAN Language Reference Manual - SINGER-Kearfott Document No. Y240A220M0101.
- 2) FORTRAN 77 Standard: "American National Standard Programming Language FORTRAN-ANSI X3.9-1978".
- 3) FORTRAN 66 Standard: "American Standard FORTRAN-ANSI X3.9-1966".
- 4) FORTRAN 77 by W. Brainerd CACM October 1978.
- 5) FORTRAN 77 by H. Katzan (Van Nostrand Reinhold Co.).
- 6) Industrial Realtime FORTRAN - Proposal of the Purdue Europe Technical Committee 1 - January 1978.
- 7) DOD Supplement to American National Standard FORTRAN -5 April 1978.
- 8) DOD Directive 5000.29, "Management of Computer Resources in Major Defense Systems", 26 April 1976.
- 9) DOD Instruction 5000.31, "Interim List of DoD Approved High Order Programming Languages (HOL)", 24 November 1976.
- 10) Selection of FORTRAN as an Avionic HOL - SINGER-Kearfott Document No. Y256A757.
- 11) A Compiler Generator by W. McKeeman, J. Horning, D. Wortman (Prentice-Hall, Inc.).
- 12) "Programming Considerations of Future Guidance & Control Computers", paper presented at AGARD Conference on The Application of Digital Computers to Guidance and Control, June 1970, by A. J. Maher.
- 13) "Parnas Partitioning" paper presented at AGARD Symposium for Techniques for Data Handling in Tactical Systems, October 1978, by A. J. Maher.
- 14) Article on "Core FORTRAN" in the FORTRAN Development Newsletter (FORWARD), Volume 4, No. 3, October 1978.

TABLE I

Feature	Language						
	FORTRAN	JOVIAL J3B	JOVIAL J73	JOVIAL J3	CMS2	TACPOL	SPL/I
Included in DODI 5000.317	Yes	No	Yes	Yes	Yes	Yes	Yes
Language Capability	Acceptable	Good	Very Good	Good	Good	Good	Very Good
Performance Simulator Commonality	Very Good	Poor	Poor	Poor	Poor	Poor	Poor
Portable Compiler Development Cost	Lowest	Modest	High	High	High	High	High
Can Compiler development be incrementally funded?	Yes	No	No	No	No	No	No
SKD Maintenance Capability	Excellent	Fair	Fair	Fair	Fair	Fair	Fair
Programmer Training Costs	None	Modest	High	High	High	High	High
Used by which Service	ALL	USAF	USAF	USAF	USN	US Army	USN

TABLE II

SKC FORTRAN VI.01
BENCHMARK RESULTS
LOCAL OPTIMIZATION IMPLEMENTED
GLOBAL OPTIMIZATION NOT YET IMPLEMENTED

		BENCHMARK #1 RANGE & BEARING	BENCHMARK #2 SEEK/COORD. CONV.	BENCHMARK #3 DIAS GN&C
IBM 370	FORTRAN G	439 (402%)	540 (284%)	2445 (384%)
IBM 370	FORTRAN HX	345 (338%)	519 (273%)	1534 (254%)
INTERDATA 8/32	FORTRAN VII	168 (165%)	370 (195%)	1058 (175%)
DEC PDP 11	FORTRAN IV +	165 (162%)	268 (141%)	963 (159%)
HP 2100	FORTRAN IV	211 (207%)	306 (161%)	1143 (189%)
SKC2000	"FORTRAN"	157 (154%)	237 (125%)	764 (127%)
SKC3121	SKC FORTRAN	102 (100%)	190 (100%)	603 (100%)

SUMMARY TABULATION OF INSTRUCTION MEMORY REQUIRED

AN OBSERVER SYSTEM FOR SENSOR FAILURE DETECTION AND ISOLATION IN DIGITAL FLIGHT CONTROL SYSTEMS

by

NORBERT STUCKENBERG
INSTITUT FÜR FLUGFÜHRUNG, DFVLR, FLUGHAFEN
D 3300 BRAUNSCHWEIG, GERMANY

Summary

For the sensor part of a flight control system a sensor failure detection and isolation concept is presented based on analytic redundancy. A conventional triplex sensor system is replaced by a duplex sensor system without loss of the fail-operational property. In the case of a sensor failure deterministic Luenberger observers provide the information about which of the two sensors of the duplex system actually failed. The proposed concept is applied to a command and stability system of a flight control system.

1. Introduction

In flight control systems a high reliability is usually achieved by using multiple devices in the vital parts of the system. An example of this well established redundancy concept is shown in fig. 1, where a triplex sensor system combined with a voting mechanism is applied for the measurement of two output signals y_1 and y_2 . In the case of a sensor failure the respective voter selects from the three sensors of either sensor type the correct signals based on a 2 against 1 decision. Hence a high probability of a correct measurement is achieved.

However it is obvious, that the concept of redundancy is expensive due to the multiplication of the sensor hardware. In order to reduce these costs, several methods have been developed, refs. (1), (2), (3), based on certain assumptions about the plant and the signals involved in the process. The general objective of all these techniques is to replace the hardware redundancy by the so-called analytic redundancy, which is realized in the processing section of a control system.

In this way for a flight control system a concept for the detection and isolation of SFs is proposed omitting the third sensor in every signal path of the triplex system of fig. 1. Nevertheless the sensor system shall keep its fail-operational capability which shall be achieved by analytic redundancy performed by deterministic observers. Special attention is directed to the problem of observer performance under the influence of unmeasurable external disturbances.

2. The system structure

Fig. 2 shows the structure of the complete system, consisting of the plant, a duplex sensor system, a controller, two observers and a logic for the detection and isolation of sensor failures (SF).

In the linear plant are

A system matrix	x state vector
B control matrix	u control vector
C output matrix	y output vector
D disturbance input matrix	z disturbance vector

The output matrix C is defined such that the resulting output vector y contains sufficient information about the state x of the plant as it is used for the control problem. The output vector y is measured by sensors which are put into a duplex configuration in such a way that two identical sensor packages SP1 and SP2 arise. Hence either SP includes one sensor of every sensor type (ST). As long as no SF occurs, the resulting two measurement vectors y_1 and y_2 are identical. A SF in a sensor of SP1 or SP2 is defined as an additive signal and represented in the failure vector v_1 or v_2 respectively.

The controller represented by the feedback matrix R is designed such that the closed loop control system gets certain desired properties, i.e. a good response to the command input u_c and a sufficient suppression of the disturbances z. Prior to a SF-isolation, which is performed in the SF-logic, the feedback loop is exclusively connected with the sensor outputs of SP2, as it is shown in fig. 2.

For the purpose of the SF-detection two deterministic Luenberger observers are separately connected with the SPs. Either observer independently generates state vectors \hat{x}_1 and \hat{x}_2 as estimates of the state vector x of the plant. For the SF-detection the innovation vectors n_1 and n_2 are used as it is indicated by the signal paths from the observers to the SF-logic in fig. 2.

With the usual applications of observers the estimated state vector \hat{x} is used as an input vector to the controller in order to improve the eigendynamic of the control system in a certain optimal sense. However, disturbances and parameter variations deteriorate the estimation performance and by that also the control performance. Therefore under those circumstances the application of observers in that way is rarely of practical use. But in the structure shown in fig. 2 the sensor outputs are directly used for the feedback loop, as it is mostly performed with flight control systems. The observers are completely separated from the control loop and exclusively used for the SF-detection.

3. Operation of the SF-logic

The SF-logic is divided in the two parts *SF-detection* and *SF-isolation*. The operation of the detection part is based on the relations between the command vector u_c , the disturbance vector z , the SF vectors v_1 and v_2 on the input side of the system and the measurement vectors y_1 and y_2 and the innovation vectors n_1 and n_2 on the output side. These relations are given by the state equations of the control system and the error systems of the two observers, which are derived from fig. 2.

Control system

$$(3.1) \quad \dot{x} = (A-BRC)x + Dz - BRv_2 + Bu_c$$

$$(3.2) \quad y_1 = Cx + v_1$$

$$(3.3) \quad y_2 = Cx + v_2$$

Error System of observer 1

$$(3.4) \quad \dot{m}_1 = (A-KC)m_1 + Dz - Kv_1$$

$$(3.5) \quad n_1 = Cm_1 + v_1$$

Error System of observer 2

$$(3.6) \quad \dot{m}_2 = (A-KC)m_2 + Dz - Kv_2$$

$$(3.7) \quad n_2 = Cm_2 + v_2$$

The error vectors m_1 and m_2 are defined as

$$(3.8) \quad m_1 = x - \hat{x}_1, \quad m_2 = x - \hat{x}_2$$

As fig. 2 shows, the innovations are used as input signals to the SF-logic. This is done due to the following properties which can be derived easily from the state equations: The innovations as output signals of the error systems are not influenced by the command input u_c , but only by the disturbances z and the SFs v_1 and v_2 . As far as the relations to the SFs are concerned, the state equations show, that the innovations n_1 of the first observer are related to SFs of the first SP only. Equivalently the innovations n_2 of the observer 2 correspond to the second SP. The effect of the disturbances on the innovations can be assumed to be finite. Hence, thresholds in the different innovation signals can be defined as their individual maximum response to disturbances.

These statements are the basis for the SF-detection scheme shown in fig. 3. There the vector signals of fig. 2 are partly transformed into a scalar representation for clarification purposes. The SF-detection is performed in the following step by step logic:

1. The output of a sensor i of the SP1 is compared with the output of the sensor i of SP2, yielding the difference signal Δy_i . When one of the two sensors of the ST i fails, the comparison shows a nonzero difference Δy_i . Thus the failed ST is detected.
2. Which of the two sensors actually failed, is identified by the above discussed effect of SFs on the innovation vectors. When one or more components of the innovation vector n_1 exceed their previously defined disturbance thresholds n_{1T} or n_{1T} respectively, it is certain that the SF is within the SP1. Equivalently a SF of SP2 is identified by the corresponding innovation signals n_{12} or n_{12} .

Thus the failed sensor is localized, since its ST and its SP are known.

When a sensor detected as failed has to be isolated, it is switched off. Its output signal is replaced by the output of the corresponding sensor of the same ST but of the respective other SP.

4. Requirements

For the application of the described SF-detection logic it has to be verified that the control system is sufficiently protected against the effects of SFs. Due to the nonzero disturbance thresholds of the innovations the SFs also can reach finite values before the failed SP is indicated by the innovations increasing beyond their thresholds. Meanwhile the SFs can have caused a deviation in the control system. Now, it has to be determined, how much offset from the nominal path due to SFs should be allowed.

The nominal path of the control system is defined by the command input $u_c(t)$. Deviations arise both from external disturbances and from SFs. From an engineering point of view the deviations due to disturbances are as undesired as they are due to SFs. Hence it is fair to say that the control system output resulting commonly from SFs and disturbances should not exceed the range defined by the maximum response to disturbances only.

The state equations derived in chapt. 3 show the SFs as input signals to both the control system and the respective error system. The effect of a SF is defined by the responses of the respective system outputs. These responses are determined by the chosen feedback matrix R for the control system and by the observer gain matrix K yet to be defined for the observer. Secondly, the responses depend on the signal character of the SFs. As far as the SF itself is concerned assumptions about its signal character are not given. Hence the SF-detection has to overcome all kind of SFs. For the observer dynamic an appropriate observer gain matrix K is required such that the innovations as output signals of the respective error system localize a failed sensor in the described way before the control system leaves its above defined range.

5. The observer gain matrix K

The state equations of chapter 3 show that the control system is affected by a SF via the feedback matrix R in the same way as the error system is via the observer gain matrix K . This dual effect of a SF on the two systems suggests to choose the observer gain matrix K such that the dynamic of the observer becomes equal to the one of the control system. Hence the matrix K is defined by

$$(5.1) \quad K = BR$$

The result of this determination shows up by the following comparison between the state equations of the two systems. But instead of applying the SF-vector v_1 or v_2 respectively as it is represented in chapt. 3, a specific SF-signal v_{12} is used representing a failure in the sensor of ST 1 and of SP2. (For simplification reasons the index 2 is omitted in the rest of the paper.)

Control system:

$$(5.1) \quad \dot{x} = (A-BRC)x - (BR)_1 v_1 + Dz + Bu_c$$

$$(5.2) \quad y_1 = C_1 x + v_1$$

$$(5.3) \quad y_j = C_j x, \quad j \neq 1$$

Error system:

$$(5.4) \quad \dot{m} = (A-BRC)m - (BR)_1 v_1 + Dz$$

$$(5.5) \quad n_1 = C_1 m + v_1$$

$$(5.6) \quad n_j = C_j m, \quad j \neq 1$$

These equations show that the state x of the control system and the state m of the error system respond equally to both the external disturbances z and a SF v_1 . By that the measurement signals y_1 and y_j and the innovation signals n_1 and n_j are identical. This result has a considerable effect on the SF-detection, which is discussed here using the time histories shown in fig. 4.

Given is the desired command response of the controlled plant output. Due to disturbances the system oscillates around this desired nominal path with a certain maximum deviation. When a SF occurs, the control system may increase beyond its disturbance range. Now, due to the special choice of the observer gain matrix, the innovation n becomes identical to the disturbance + SF response of the control system. That means first, the maximum deviation of the control system due to disturbances can be defined as disturbance threshold for the innovation. Secondly when a SF drives the control system beyond an unusual high deviation from the nominal path, the innovation exceeds its disturbance threshold as well, by that indicating the failed sensor. On the other hand, if the SF-influence on the control system is low, then there is no need for an indication. By that all kind of SFs are covered as far as their influence on the control system is severe.

6. Critical failure frequencies

The result of the previous chapter is based on the identity between the outputs of the control system and of the error system. Thus the measurement signals y_i and y_j do not exceed their disturbance range unless the innovations localize the SF. Hence the control system seems to be sufficiently protected against SFs.

However it is not the measurement signal which has to be protected but rather the plant outputs represented as $C_i x$ and $C_j x$ in equ. (5.2) and equ. (5.3). For the nondiagonal relation $C_j x/v_i$ and n_j/v_i with $j \neq i$ the discussed identity still holds. However in the diagonal relation $C_i x/v_i$ and n_i/v_i the identity is disturbed, as it is indicated by the additive failure signal v_i in the output equation (5.5). Under certain circumstances this happens possibly in a way, that in equ. (5.5) a compensation of the SF v_i and its response $C_i m$ takes place. By that a SF v_i would drive the plant output $C_i x$ further than it would be indicated by the diagonal innovation n_i . To discuss this problem, the relations between the SF v_i and the plant output $C_i x$ on the one hand and the innovation on the other hand can also be expressed in an equivalent relation where the innovation n_i is defined as a new input signal. The corresponding state equations are:

$$(6.1) \quad \dot{m} = (A - BRC + (BR)_i C_i) m - (BR)_i n_i$$

$$(6.2) \quad C_i m = C_i x$$

$$(6.3) \quad C_j m = C_j x = n_j, \quad j \neq i$$

For a given system it has now to be examined if there exist input signals n_i which generate output signals $C_i x > n_i$. This can easily be done in the frequency domain with the corresponding transfer function $C_i x/n_i(\omega)$. If the amplitude plot $|C_i x/n_i|(\omega)$ shows a configuration like that in fig. 5 with values $|C_i x/n_i| > 1$, then SFs v_i synthesized with frequencies from the critical range compensate their effects $C_i m$ in equ. (5.5) as mentioned above.

The critical frequency range of fig. 5 is expected to cover the vicinity of the eigenfrequency of the system defined in equ. (6.1). This case actually exists, if the sensor of ST i fails completely, i.e. the sensor generates a constant signal independently of the system state x . This failure case is the wellknown hardover failure. Hence the problem of the critical frequency range is not at all arbitrary.

This problem can be overcome if a coupling from the diagonal output $C_i x$ to a nondiagonal $C_j x$ exists in the hardover system defined in equ. (6.1). In this case a SF v_i cannot drive the output $C_i x$ too far away from the desired path unless the output $C_j x$ and the innovation n_j increase as well. Therefore the failed SP is identified eventually by the innovation n_j .

For cases where the SF-detection performed by the nondiagonal innovation n_j is still too poor, i.e. the plant output $C_i x$ deviates too far from its nominal path, an improvement of the SF-detection is proposed, as it is shown in fig. 6. A filter is connected at the nondiagonal innovation n_j and specially matched to the critical frequency of the SF v_i . The bandwidth of this filter has to cover the critical frequency range defined in fig. 5. Hence it is called "hardover filter". After the disturbance threshold of the filter output n_{jF} is determined, the indication of the failed SP is run equivalently to the procedure described in chapt. 3.

However the application of the hardover filter is limited to systems where a sufficient high coupling from the plant output $C_i x$ to another $C_j x$ actually exists. This coupling has to be an inherent property of the hardover system.

7. Example

The presented failure detection and isolation system is applied to a flight control system for the lateral motion of the executive jet "HFB 320". The state vector y , the control vector u and the disturbance vector z are defined as follows:

$$x = \begin{bmatrix} \text{side slip angle } \beta \\ \text{roll rate } p \\ \text{yaw rate } r \\ \text{bank angle } \phi \end{bmatrix}; \quad y = \begin{bmatrix} p \\ r \\ \phi \end{bmatrix}; \quad u = \begin{bmatrix} \text{aileron defl. } \xi \\ \text{rudder defl. } \zeta \end{bmatrix}; \quad z = \begin{bmatrix} \text{side slip angle gust } \beta_g \\ \text{roll rate gust } p_g \\ \text{yaw rate gust } r_g \end{bmatrix}.$$

The plant matrices are:

$$A = \begin{bmatrix} -0.11 & 0.00 & 0.99 & -0.15 \\ 0.90 & -1.09 & 0.47 & 0 \\ -2.03 & -0.12 & -0.15 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}; \quad B = \begin{bmatrix} 0 & -0.03 \\ -3.27 & 0.62 \\ -0.24 & -0.85 \\ 0 & 0 \end{bmatrix}; \quad D = \begin{bmatrix} -0.11 & 0.00 & 0.99 \\ 0.90 & -1.09 & 0.47 \\ -2.03 & -0.12 & -0.15 \\ 0 & 0 & 0 \end{bmatrix}$$

The chosen control law is:

$$u = -Ry \text{ with } R = \begin{bmatrix} 0 & -0.4 & -0.8 & -1.0 \\ 0 & 0 & -1.5 & 0 \end{bmatrix}$$

According to the presented concept the observer gain matrix is chosen as:

$$K = BR = \begin{bmatrix} 0 & 0.05 & 0 \\ 1.31 & 1.68 & 3.27 \\ 0.10 & 1.47 & 0.24 \\ 0 & 0 & 0 \end{bmatrix}$$

Fig. 7 shows the structure of the simulation arrangement. Hardware sensors are included in the closed loop in order to test the system in a more realistic environment than it is possible with a mere software simulation. For this reason two rate gyros for the roll and yaw rates and one attitude gyro for the bank angle are mounted on a gyro test bed. This SP represents the SP2 as it is introduced in fig. 2. The test bed itself is driven by the roll rate p and the yaw rate r as output signals of the plant. The rest of the system simulated in a process control computer corresponds to the structure shown in fig. 2.

The control system is designed as a command and stability system. Thus it has a good response to the command signal given as the commanded bank angle. Furthermore an acceptable suppression of the disturbances defined as gusts is achieved.

Fig. 8 shows the response of the three plant outputs p , r , ϕ after a step input in the bank angle command ϕ_c . These time histories are used as references when SFs are applied in some of the next figures. All innovations n_i and sensor output differences Δy_i used as input signals to the SF-logic are close to zero yet, apart from a low noise level in the sensors.

In fig. 9 disturbances are applied to the system. Since the observer gain matrix K is chosen in the described special way, the response of the controlled plant is identical to the corresponding innovation signals. The maximum values define the respective disturbance thresholds of the innovations. In order to receive the maximum disturbance responses, thunderstorm gusts (Dryden power spectr., MIL-F-8785) are applied. Due to these disturbances the innovations do not exceed the thresholds $n_{rT} = 6^\circ/\text{s}$, $n_{rT} = 5^\circ/\text{s}$, $n_{pT} = 4^\circ$. The sensor output differences are still at zero due to the fact that no SF has occurred up to now.

Failure case 1:

In fig. 10 a zero shift failure in the yaw rate sensor is applied, together with a step input in the bank angle command ϕ_c . The response of the control system signals and the detection signals are shown, on the left side without an isolation of the failed sensor and on the right side with isolation. The zero shift in the yaw rate sensor causes a deviation of the bank angle essentially, apart from a slight influence on the roll and yaw rates. The detection of the failed sensor is performed in the described two stages. First, the sensor output differences of the two yaw rate gyros shows a distinct signal at the time of SF-occurrence. Hence, the type of the failed sensor is detected. Secondly, all three innovations of the observer 2 exceed their previously defined disturbance thresholds. Therefore, it is certain that the failed sensor is located in the second SP. The SF-detection is completed. With the activated sensor isolation, on the right side of the picture, the control system recovers to the desired values.

Failure case 2:

A SF of the critical kind discussed in chapt. 6 is applied to the bank angle sensor of the SP2. An investigation of the corresponding hardover system defined in equ. (6.1) shows an eigenvalue close to zero. That means, the critical SF v_ϕ has a stationary signal character. Thus the critical SF in the bank angle sensor is drift. Due to the structure of the hardover system the bank angle ϕ is following the false measurement in such a way that a compensation discussed in chapt. 6 occurs in the equ. (5.2) and equ. (5.5). Hence the corresponding innovation n_ϕ does not indicate the drifting SF v_ϕ . Since in the hardover system a small coupling from the innovation n_ϕ to the innovation n_r still exists, the concept of the hardover filter can be used. In this special case the proposed hardover filter connected with the innovation n_r has to be sensitive with respect to stationary signals. Hence a low pass filter is chosen.

Fig. 11 shows the corresponding amplitude plot of a low pass filter with a time constant $T = 20$ s. Due to the high gain at low frequencies and low gains over the rest of the frequency range the filter output n_{rF} is expected to show a distinct response when the system is forced by a drifting SF v_ϕ . Disturbances however, represented by the flat power spectrum $S_{zz}(\omega)$ will not cause an equivalent response of the filter output n_{rF} .

Fig. 12 and fig. 13 show the results in terms of time histories. The maximum value of the filter output n_{rF} due to the previously defined thunderstorm gusts is $n_{rFT} = 0.4^\circ/\text{s}$.

Fig. 13 shows the relevant signals of the system when it is forced by a SF v_ϕ with a drift rate of $0.1^\circ/\text{s}$. Prior to the SF-isolation the bank angle ϕ follows the SF v_ϕ such that in the output equ. (5.5) of the corresponding error system the innovation stays at zero. However, due to the described coupling the innovation n_r is drifting, too. By that the hardover filter output n_{rF} reaches its threshold n_{rFT} and identifies the failed SP. Then

the failed bank angle sensor is isolated. The control system recovers without the bank angle ϕ itself having exceeded its admissible range.

8. Conclusions

For the sensor part of a flight control system a failure detection mechanism has been derived based on analytic redundancy. With a duplex sensor configuration the fail-operational properties of a triplex system are achieved. In the case of a sensor failure the innovation signals of two deterministic observers provide the information for the logical decision about which of two sensors have failed actually. The observer gain matrix is chosen in such a way that the observer dynamic becomes equal or close to the control system dynamic. This is done in order to cover all kind of sensor failures as far as their influence on the control system is severe. In certain cases, depending on the structure of the system, an additional filter has to be applied to overcome critical failure frequencies. This method is successful as far as the failure signal still affects other outputs of the control system.

Hardware sensors included in the closed loop of the simulation arrangement did not deteriorate the results achieved by the described failure detection method.

9. References

1. Shapiro, E.Y.: "On the availability of redundant signal information for aircraft control in the presence of sensor failures", Proc. of the 1976 conf. on inf. sciences and systems, Baltimore, USA.
2. Cunningham, T.B., Poyneer, R.D.: "Sensor failure detection using analytic redundancy", Proc. of the 1977 JACC.
3. Clark, R.N. et al.: "Detecting instrument malfunctions in control systems", IEEE transact. on aerospace and electronic systems, Vol. AES-11 No. 4, July 1975.

FIGURES

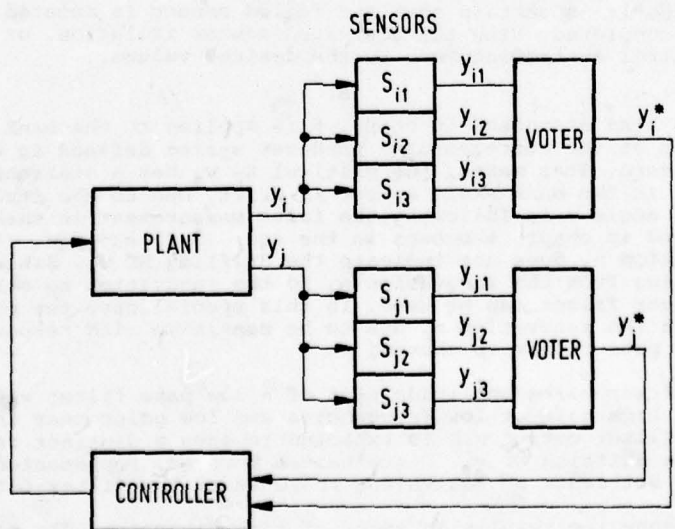


Fig. 1: Control system with a conventional triplex sensor configuration

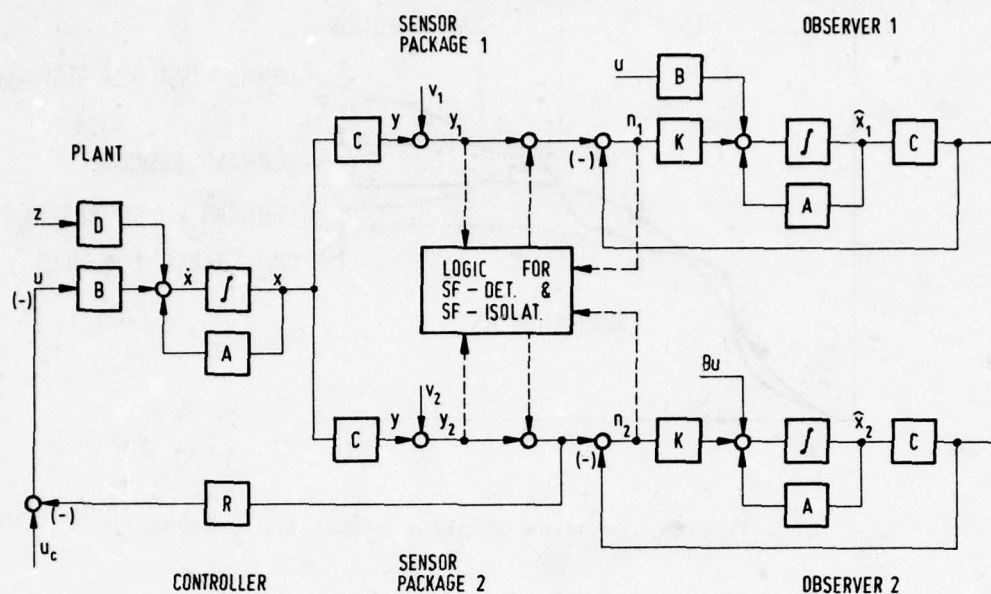


Fig. 2: Control system with a duplex sensor configuration and observers for analytic redundancy

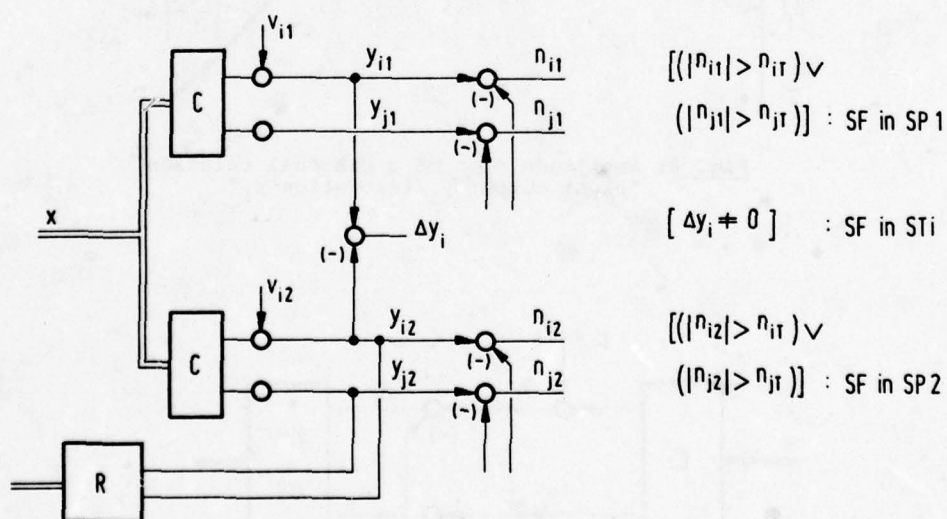


Fig. 3: Logic for sensor failure detection

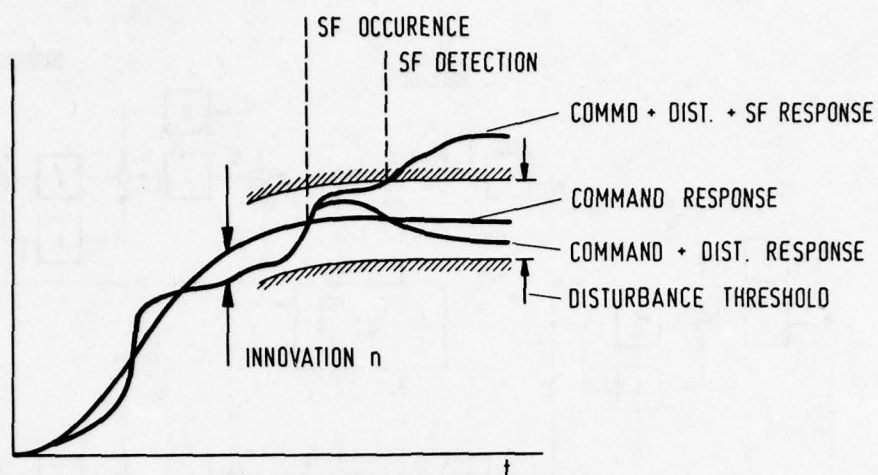


Fig. 4: Time histories of plant output and observer innovation

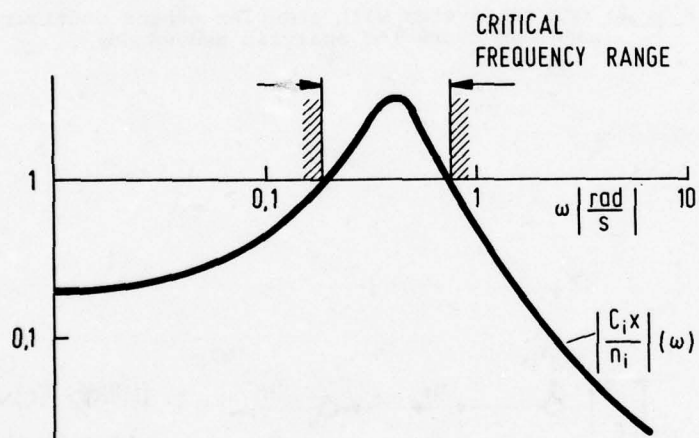


Fig. 5: Amplitude plot of a diagonal relation "plant output y_i /innovation n_i "

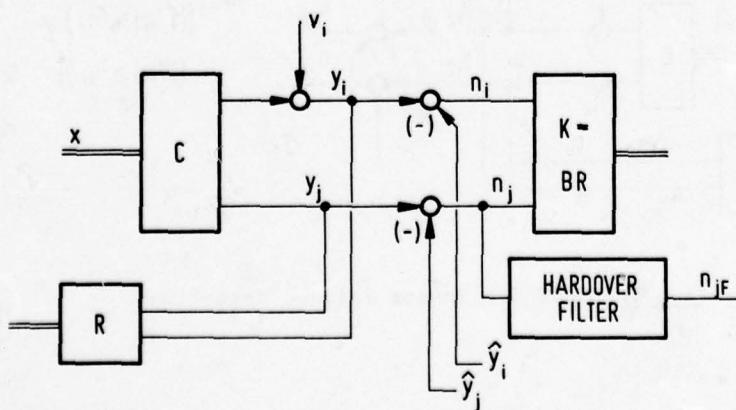


Fig. 6: Hardover filter matched to the failure signal v_i

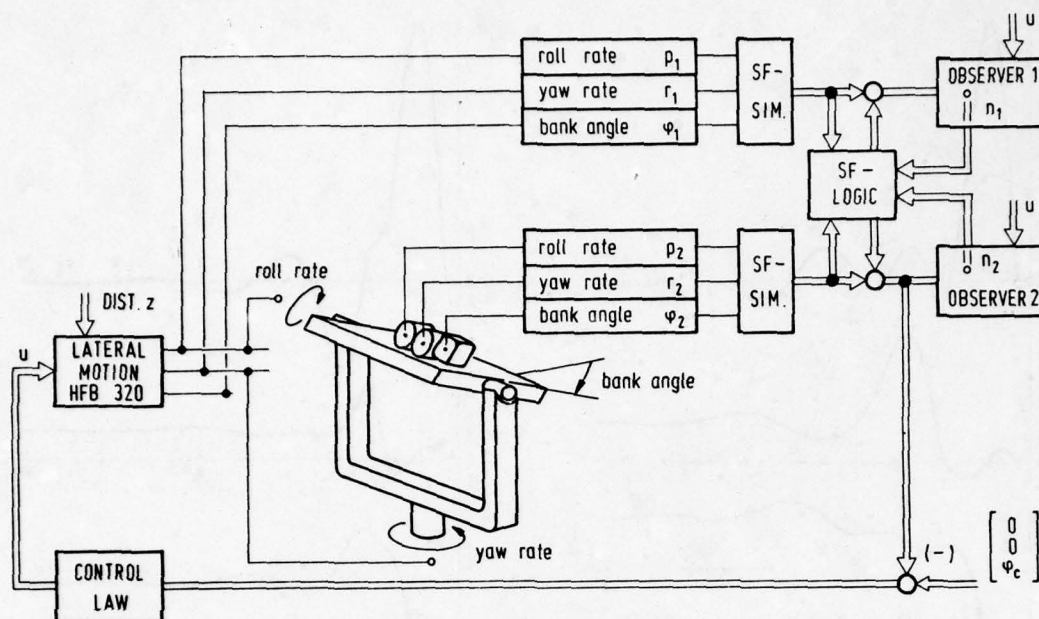


Fig. 7: Simulation arrangement for tests with a CSAS

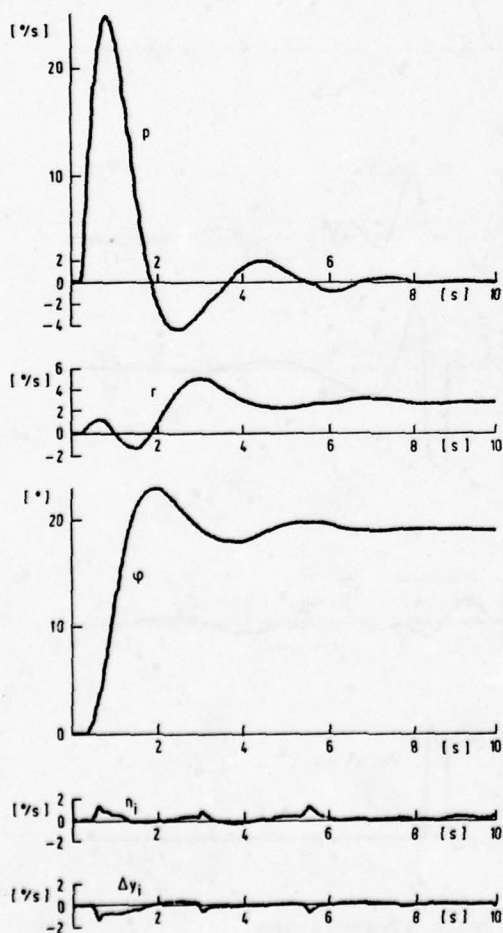


Fig. 8: Response to a bank angle command

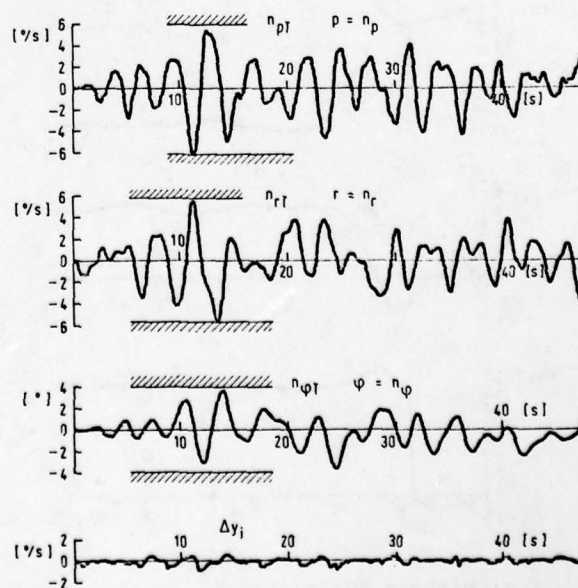


Fig. 9: Gust responses (Dryden power spectr., (MIL-F-8785))

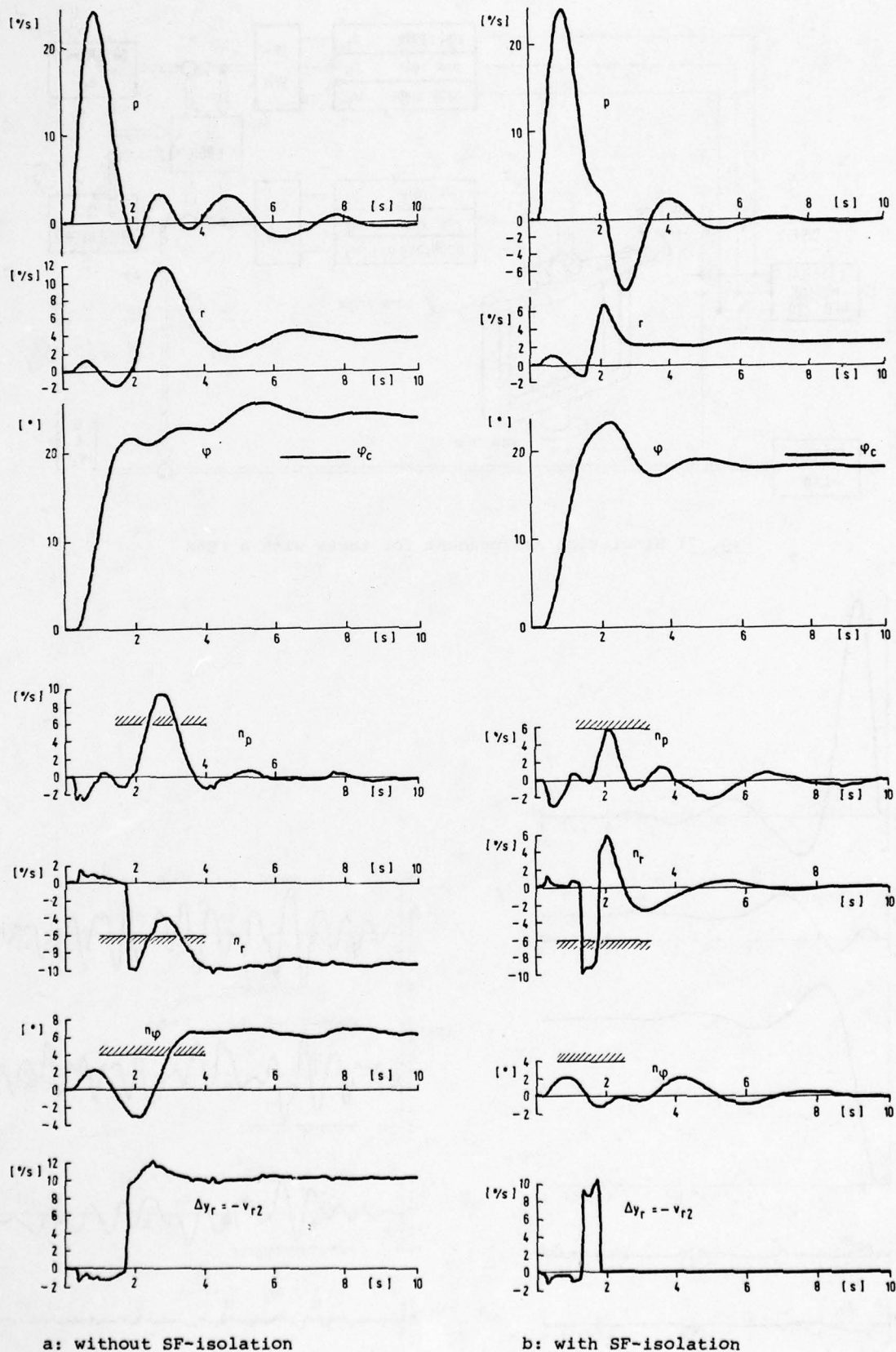


Fig. 10: Detection and isolation of a zero shift failure in the yaw rate sensor

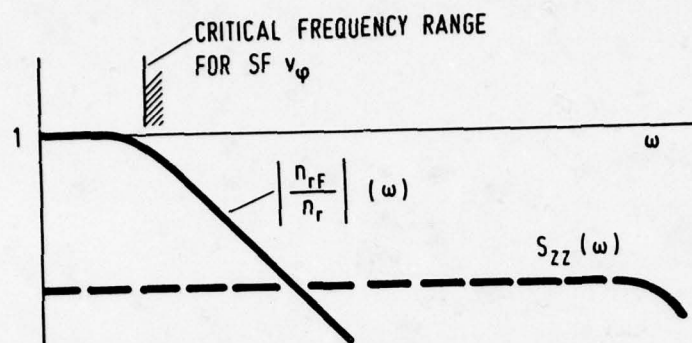


Fig. 11: Amplitude plot of the hardover filter $(n_{rF}/n_r)(\omega)$

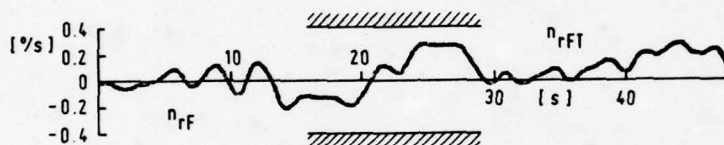


Fig. 12: Gust response of the hardover filter output n_{rF}

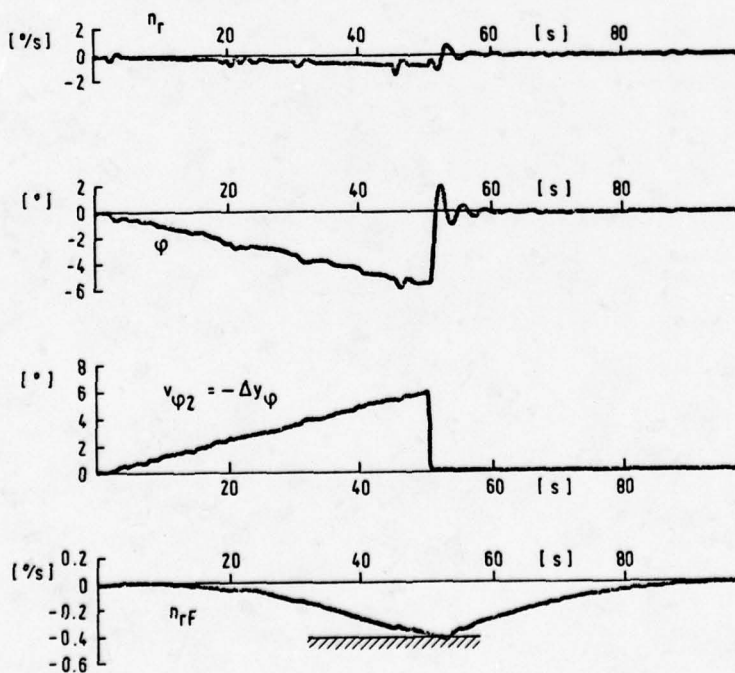


Fig. 13: Detection and isolation of drift in the bank angle sensor (drift rate $\dot{v}_\phi = 0.1^\circ/s$)

AUTOMATIC RECOVERY AFTER SENSOR FAILURE ON BOARD

by

Marc Labarrère, Marc Pélegrin, Marc Pircher

O N E R A - C E R T *

2 Avenue Edouard Belin, 31000 TOULOUSE, FRANCE

SUMMARY

Two techniques are developed which provide reliable failure detection and isolation for a dual-redundant subset of sensors. The paper starts with a global procedure using a bank of stationary Kalman filters. Some outlined difficulties of this technique lead to a sub-optimal procedure which is developed in order to give all the dynamic and static relationships between the measured outputs on the aircraft.

These techniques are successfully applied to simulated sensor failure on a six degree of freedom aircraft simulation and are applying to sensor failures injected on flight data from the N262 aircraft.

INTRODUCTION

The most attractive applications of CCV (control - configured - vehicle) concept - certificial longitudinal stability, combined with automatic configuration management - requires a full-time, full-authority control system for its implementation.

This implies a very high reliability of the essential sensors, computers, actuators and their associated power supplies, for flight safety.

In order to achieve this reliability, current practice for fault tolerant operation involves triplex or quadruplex redundancy followed by a voting system (cross - channel voting). Such a system costs a lot of weight, power, size and money.

The introduction of on-board digital computer presents the possibility of reducing the redundancy level from quadruplex to triplex (even in some cases, duplex).

This paper is not dealing with the fields of Fault - tolerant computer and actuators failure, but it presents two redundancy management techniques in order to reduce the number of sensors required to maintain undegraded performance when two non simultaneous failures occur in a triplex system, or one in a duplex system (even two in some cases).

Then, the problem is to be highly reliable after a single failure in a system in which only two sensors of each type are present.

The techniques in essence involve voting, but the third information is provided by analytic redundancy which exists in dynamic relationships between the different outputs measured on the aircraft.

Since the plant is well known and the information from sensors is contaminated by noise and turbulence, the creation of the third information is inherently an estimation problem.

The problem is essentially algorithmic, and many interesting estimation algorithms have been developed so far [1 - 10]. But many try to give the "best estimate" under restrictive assumptions (without turbulence in some cases or open loop system) and do not design a control system that tolerates the failure of an instrument whether or not its detection has taken place.

At C.E.R.T./DERA in a previous study, we have investigated the possibility of replacing the third sensor by an estimation obtained through a set of pre-computed Kalman filters. In order to overpass certain difficulties, discussed further, the authors have developed a new procedure which is based upon a partition of the system in sub-systems but the main ideas remain the same.

I. FORMULATION OF THE PROBLEM

We work with two independent sets of measurements S' and S'' of output vector with the corresponding "normal" noises W' and W'' with known statistics.

- the computer and the actuators are operational and not subjected to failure
- the airplane has two kinds of inputs, well known control law U delivered by the auto-pilot and unknown disturbances such as turbulence.

* Office National d'Etudes et de Recherches aérospatiales, Centre d'études et de recherches de Toulouse

- we have developed our failure detection techniques for the sensors on the N262 and the AIRBUS aircrafts, although their application to another aircraft with a different sensor set is straightforward.
- the system is subjected to sensor failures such as abnormal offsets, drifts, saturations, jumps, amplified noises, gain variations ... These failures appear randomly and their effects are not always additive.

Under these assumptions, the problem is not to detect the failure but to choose the valid sensor. The principle of the detection and isolation is shown on figure 1.

II. ESTIMATION - TECHNIQUE

The Kalman filter gives for this problem a general solution taking advantage of all the information about the system, but it assumes that the system and the perturbation are well known and well described by the state equations (1)

II.1 - KALMAN filter equations :

Let us consider the following system :

$$\begin{cases} X_{n+1} = AX_n + BU_n + V_n \\ Z_n = CX_n + DU_n + W_n \end{cases} \quad (1)$$

where :

X_n is the state vector at time $t = t_n$

Z_n is the output vector

U_n is the vector of known inputs

V_n, W_n are white independent sequences, with zero means and covariance matrices Q_n and R_n .

A, B, C, D are the matrices of the state equations of the systems.

Let $X_{n1/n2}$ the estimate of the state vector X at time t_{n1} knowing all the outputs $Z_0 \dots Z_{n2}$

The equations of the filter give the best linear estimate $X_{n/n}$ of X_n and the covariance matrix $P_{n/n}$ of the estimation error :

$$X_{n+1/n} = AX_{n/n} + BU_n$$

$$X_{n+1/n+1} = X_{n+1/n} + K_{n+1} (Z_{n+1} - Z_{n+1/n})$$

$$Z_{n+1/n} = CX_{n+1/n} + DU_{n+1}$$

$$P_{n+1/n} = AP_{n/n}A^T + Q_n$$

$$K_{n+1} = P_{n+1/n}C^T (CP_{n+1/n}C^T + R_{n+1})^{-1}$$

$$P_{n+1/n+1} = (I - K_{n+1}C) P_{n+1/n} (I - K_{n+1}C)^T + K_{n+1} R_{n+1} K_{n+1}^T$$

with : I unit matrix
 T transposition symbol

II.2 - Adaptation to failure detection

Knowing the state of the system, we are able to compute an estimate of each output :
 $Z_{n/n} = CX_{n/n} + DU_n$ which can be used to test the Z_n information.

The system with the two sets of measurements is described by the equations :

$$\begin{cases} X_{n+1} = AX_n + BU_n + V_n \\ S'_n = CX_n + DU_n + W'_n \\ S''_n = CX_n + DU_n + W''_n \end{cases}$$

V_n, W'_n, W''_n are independent white sequences with zero means and covariance matrices Q_n, T'_n, T''_n .

Without any failure, all the measurements are used and the observation vector is

taken as follows :

$$z_n = \frac{S'_n + S''_n}{2} \quad \text{and its noise covariance matrix } R_n = \frac{T_n}{2}$$

The detection is made by comparing the components of the magnitude of $|S'_n - S''_n|$ with a set of detection thresholds. If the i^{th} threshold is surpassed, the two measurements of the halfsum must be suppressed.

Obviously, in presence of failure, one Kalman filter running with all the outputs, cannot solve the estimation problem, because the behaviour of the estimate can be incorrect. Such a confusion would give bad estimates unable to make a choice between two incoherent measurements.

But a bank of stationary filters give a good solution if each filter is well adapted to a particular case.

Assuming that simultaneous failures occurring on different type of sensors are possible, we need :

$$C_m^{m-1} + C_m^{m-2} + \dots + C_m^1 + 1 = 2^m - 1$$

filters working simultaneously, if m is the number of measured outputs.

The calculations can be reduced if :

- there are not two different failures occurring at the same time
- switching from one filter to another replaces the parallel calculation.

With this solution $C_m^{m-1} + 1 = m + 1$ gain matrices corresponding to $m+1$ filters are needed and have to be stored.

- a matrix $K^{(0)}$ which uses all the outputs in order to avoid the switch transient by initializing the other effects.
- m matrices $K^{(i)}$ which use all the outputs except the i^{th} one.

This design is summarized in figure 2. The logical procedure of choice is as follows :

- . $|S'_n(i) - S''_n(i)| < \epsilon_i$ for any $i = 1, m$ we use the filter with the gain matrix $K^{(0)}$
- . For the i^{th} output $|S'_n(i) - S''_n(i)| > \epsilon_i$ both components $S'_n(i)$ and $S''_n(i)$ are eliminated and the gain matrix $K^{(i)}$ is chosen. Then the i^{th} filter delivers the estimate $z_{n/n}^{(i)}$ without using the unsafe measurement, and the choice of the correct sensor can be performed. This procedure is shown on Figure 3.

Under the assumptions given above, this estimation technique was efficient to solve the problem of failure detection on the AIRBUS A 300 B2 (reference 8)

The following remarks lead to the new concept which provides fault detection and isolation through analytical redundancy.

- The Kalman filter gives an output estimate $z_{n/n}$ through the estimate state vector $X_{n/n}$ which includes the wind.

Therefore, the turbulence has to be characterized by a statistical model, such as Dryden form; hence the filter provides an estimation of the wind. But for sensor failure detection purpose, all we need is some relations between the outputs of instruments which are being subjected to the same inputs and then the knowledge of all the state estimates is not necessary. The accuracy of the estimates is function of the model accuracy.

- This global procedure does not tolerate another failure during the interval of time between the detection of the first failure and the rejection of the faulty sensor. In this case, the effect of the new failure is the same that a wrong decision or simultaneous failures, the filters work with incorrect information and the estimates are biased and then the overall procedure becomes unsafe.

- Since the Kalman filter minimizes the covariance matrix $P_{n/n}$ of the estimation error, with respect to the "known" statistics of the measurement noises and the plant perturbations (turbulence) it can neglect a sub-optimal relation independent of the flight conditions or of the wind model (such as the relationship between the Euler angles and their rates) which could be better for failure detection purpose.

Then, the initial goals for the new concept are as follows :

- try to be free from the statistical model of turbulence and the nominal flight condition.

- in order to reduce the consequences of a wrong decision and simultaneous failures, remove the global procedure to sub-structures used in a parallel mode.
- then, find the minimum number of outputs involved in a dynamic as static relationship.
- . the control law U computed by the autopilot must be obtained from safe measurements
- . design simplicity to insure minimum on board computer requirements

III. ANALYTICAL REDUNDANCY

III.1 - Procedure

This concept is first based upon researching all the dynamic or static relationships between all the subgroups of the output vector of a dynamic system with unknown inputs.

The aircraft dynamic and measurement equations can be described as follows :

$$\begin{cases} \dot{X} = AX + BU + EV \\ Z = CX + DU + FV \end{cases}$$

where :

X is the $n \times 1$ state vector ($\dot{X} = \frac{dX}{dt}$)

U is the 1×1 known input vector

V is the $q \times 1$ unknown input vector

Z is the $m + 1$ output vector, so far the measurement noises are not introduced

We are looking for first order relationships between the outputs and their derivatives that is to say $MZ + N\dot{Z} = 0$ where M and N are constant matrices. These relations have to be satisfied in the absence of failure.

We have :

$$\begin{bmatrix} \dot{X} \\ U \\ \dot{Z} \\ Z \end{bmatrix} = \begin{bmatrix} C & D & F & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ - & - & - & - & - \\ CA & CB & CE & D & F \\ 0 & 0 & 0 & I & 0 \end{bmatrix} \times \begin{bmatrix} X \\ U \\ V \\ V \\ V \end{bmatrix}$$

or, $Y = \mathcal{C}X$ where \mathcal{C} is a $2(m+1) \times n + 2(1+q)$ matrix.

Then a modified Gauss - Jordan method is used in order to give all the linear dependences between the rows of matrix \mathcal{C} . When $(2m)$ is greater than $(n + 2q)$ it is obvious that there are at least $2m - n - 2q$ relations of dependences between the outputs and the known inputs of the system. When all the rows are independent, a higher order differentiation is used.

This procedure delivers a set of redundancy relations useful for failure isolation. The static relations can be applied directly in the isolation procedure. For the dynamic ones, the derivative \dot{Z} is not available, so far in the continuous case, there are several possibilities :

- Estimate the derivatives using high pass filters in order to reduce the amplification of high frequency noises.
- Integrate the relation and then compute the integral of the measurements. This solution leads to divergence in the presence of biases.
- Filter the overall relations through the same first order filters, thus the failures are also filtered and will not be detected.

We have preferred to compute the doubtful output - from the others and their estimated derivatives; Figure 4 shows the principle of this technique on an example.

In other words, this algorithm is modified in order to deliver several matrices H such as $Y = HY$ where H has zeros on the first diagonal.

For failure isolation purpose, the remaining problem is to choose the matrix H out of the others which will be used to compute the estimate \hat{Y} from the vector of pseudo measure :

$Y_m : \hat{Y} = H Y_m$ where $Y_m = \begin{bmatrix} Z \\ U \\ \dot{Z} \\ U \end{bmatrix}$

Z is just the sensor outputs vector,
 U is the known inputs vector delivered by the auto-pilot,
 and the differentiation is achieved through high pass filters.

The important characteristic of this choice is to find an optimum with the following constraints.

- Minimum number of outputs involved in a relation, then maximum number of zeros in the rows of matrix H . Furthermore, when the impact of one output in a relation is less than the size of the noises, the corresponding terms is cancelled out.
- The reconfiguration relations (rows of H) must be decoupled, in other words, if the i^{th} output is reconfigured from the j^{th} one, the j^{th} relation does not have to use the i^{th} output. Therefore, the columns of matrix H must have the maximum number of "zeros".
- Each component of vector Y is contaminated by noise (measurement noise), then we take the minimum weight relation taking care of all the noise statistics.
- From the aeronautical point of view, the variations of the H terms due to modification of flight conditions (variation of matrices A, B, C, D, E, F) must be kept to a minimum.

This technique does not give the states of the aircraft and the unknown inputs (turbulence) V but it eliminates the effect of perturbations on the aircraft.

Let see an example : a dynamic system with two outputs such as :

$$\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} a_1 & \dots & a_p \\ 2a_1 & \dots & 2a_p \end{bmatrix} \times \begin{bmatrix} X \\ U \\ V \end{bmatrix} + \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}$$

Obviously there is a static relationship between Z_1 and Z_2 : $Z_2 - 2 Z_1 = \epsilon$ where ϵ is function of the measurement noises W_1 and W_2 . In this trivial case, we get :

$$\begin{bmatrix} \hat{Z}_1 \\ \hat{Z}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1/2 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}$$

which is a very simple relation to implement, instead of a solution given by a technique which would try to give Z through the estimation of X and V , assuming some statistical properties) such as Kalman filter or even least square method.

Remark : when the effects of plant perturbations (gust) are neglected (E and F equal zero) the algorithm gives, among the relations, the plant equations as dynamic relationships between the outputs when the states X are measured. For example, it leads to the lift and drag equations of the aircraft equations of motion.

In presence of perturbations, we get the "best" relations with respect to the above multicriterion. The observability of the states of the system is not a necessary condition for the obtainment of relationships between the outputs.

III.2 - Failure detection procedure

The detection is made as we suggested in section III.2 - from detection thresholds ϵ_1 , but at every step (k) four pieces of information are available on each measure (i) :

- the two outputs from alike sensors $S'_k(i)$ with measurement noises $W'(i)$ and $W''(i)$
- the roughly predicted value $\hat{Z}_k(i) = Z_{k-1}(i) + Z_{k-1}(i)\Delta t$ where $Z_{k-1}(i)$ is the unfailed output at the previous step.

$Z_{k-1}(i)$ is either $S'_{k-1}(i)$ or $S''_{k-1}(i)$ depending upon the following test :

$$|S'_{k-1}(i) - Z_{k-1}(i)| < |S''_{k-1}(i) - Z_{k-1}(i)| \quad \text{then} \quad Z_{k-1}(i) = S'_{k-1}(i)$$

$$|S''_{k-1}(i) - Z_{k-1}(i)| < |S'_{k-1}(i) - Z_{k-1}(i)| \quad \text{then} \quad Z_{k-1}(i) = S''_{k-1}(i)$$

- The estimated value $\hat{Z}_k(i)$ at time $k\Delta t$:

$$\hat{z}_k^i = \sum_{j=1}^{n+2(p+q)} h_{ij} y_{mk}^{(j)} \quad \text{with } h_{ii} = 0$$

where the h_{ij} are the coefficients of matrix H , and $y_{mk}^{(j)}$ is the vector defined in section IV.1. The control laws U_k must be computed from safe measurements. It is the purpose of the predicted value \hat{z}_k by taking the closest measurement from \hat{z} as input of the autopilot and to create $y_{mk}^{(j)}$ we never take an erroneous output, dangerous for flight safety.

When a failure occurs $|S_k^{(i)} - S_k^{''(i)}| > \epsilon_1$, as soon as one of the incoherent measures $S_k^{(i)}$ or $S_k^{''(i)}$ is far from $\hat{z}_k^{(i)}$ with a larger value than the rejection thresholds δ_i , the corresponding sensor is considered to be out of order. As long as the remaining sensor output is coherent with its corresponding estimate, it can be kept to reconfigure the other outputs. So far the sub-structure concept is very useful, even a wrong decision or doubled failure on alike sensors do not affect the overall procedure.

IV. APPLICATION

This last method is studied for the N.262 aircraft and its longitudinal set of sensors : inertial sensors, accelerometers, rate gyros, aerodynamic sensors such as alpha vane and aerodynamic speed (pitot tube).

The matrices A , B , C and D are derived from linearization of the equation of motion and, hence, they are function of the design flight conditions; so seems to be the matrix H , but it is not a very restrictive condition for the obtained dynamic relations. As we have seen in a sensitivity study with various flight conditions, the principal varying terms such as dynamic pressure are cancelled out in the final relations.

The next section deals with the effects of turbulence and the derivation of the matrices E and F .

IV.1 - The Turbulence

The key problem in the design of in-flight sensor failure detection and identification is the unknown inputs of the system : turbulence and wind shear.

The most convenient way to introduce the effect of turbulence into the airplane equation of motion, and hence, into the measurement vector, is by velocity components of turbulence : U_v, V_v, W_v in the inertial frame R_0 defined at the airplane center of gravity G .

Then, we have the following rotations between the three frames R_s (G, X_s, Y_s, Z_s) $R(G, X, Y, Z)$ and R_a (G, X_a, Y_a, Z_a) where R is the rigid body frame and $G X_s$ and $G X_a$ are made respectively parallel to the ground speed V_s and the airspeed V_a :

<u>Frames</u>	<u>Rotations</u>
$R \longrightarrow R_a$	$\alpha_a GY + \beta_a GZ_a$
$R \longrightarrow R_s$	$\alpha GY + \beta GZ_s$
$R_a \longrightarrow R_s$	$\alpha_J GY_a + \beta_V GZ_s$

From $R \supset R_a \supset R_s \equiv R \supset R_s$ we get using the conventional small angle approximations.

$$\begin{cases} \alpha = \alpha_v + \alpha_a \\ \beta = \beta_v + \beta_a \end{cases}$$

Since the aircraft speed \vec{V}_s with respect to the ground is the sum of the air speed \vec{V}_a and the turbulence \vec{V}_v

then,

$$\begin{bmatrix} V_s \\ 0 \\ 0 \end{bmatrix}_{R_s} = \begin{bmatrix} V_a \\ 0 \\ 0 \end{bmatrix}_{R_a} + \begin{bmatrix} U_v \\ V_v \\ W_v \end{bmatrix}_{R_0}$$

with the rotations defined above and again small angle approximations, we have the incremented angles of attack and sideslip due to continuous turbulence :

$$\alpha_v = \frac{W_v}{V_a}$$

$$\beta_v = \frac{V_v}{V_a}$$

$$V_a = V_I - U_v$$

Since W_v and V_v have some distribution over the length and span of the aircraft, so also do α and β_v which are not necessarily uniform over the airplane because of the angular velocities (p , q and r). In accordance with Etkin [13] the variation of W_v over the length of the airplane evaluated at the c.g, is equivalent aerodynamically to the inertial pitching velocity q , similar reasoning leads to the following expressions :

$$p_v = \frac{\partial W_v}{\partial y}, \quad q_v = - \frac{\partial W_v}{\partial x} = - \alpha_v \quad \text{and} \quad r_v = \frac{\partial V_v}{\partial x}$$

Turbulence increments are applied to the equations of motions only through the aerodynamic terms, and are not applied to inertial terms in the equations. Then the aerodynamic terms (such as C_{mq}) in the equations of motion that involve the angular velocity components are simply multiplied by the difference of the inertial and turbulence angular velocities :

$$p_a = p - p_v$$

$$q_a = q - q_v$$

$$r_a = r - r_v$$

For the longitudinal motions, the effects of U_v , α_v and q_v lead to the matrices we need E and F.

In order to see the effects of wind on the measurement vector, we simulated a non linear six degrees of freedom aircraft, flying in a turbulence (u_v , v_v , w_v).

For simulation purpose, we took the Dryden spectral forms for the spectral densities of u_v , v_v and w_v :

$$\phi_{u_v}(\omega) = \tau_u^2 \frac{2 L_u}{\pi} \frac{1}{1 + (L_{u\omega})^2}$$

$$\phi_{v_v}(\omega) = \tau_v^2 \frac{L_v}{\pi} \frac{1 + 3(L_v \omega)^2}{(1 + (L_v \omega)^2)^2}$$

$$\phi_{w_v}(\omega) = \tau_w^2 \frac{L_w}{\pi} \frac{1 + 3(L_w \omega)^2}{(1 + (L_w \omega)^2)^2}$$

These turbulences are generated by white noises passing through linear filters.

There are direct effects of the gust on the aerodynamic sensors, such as angle of attack vane which senses $\alpha = \alpha - \alpha_v$; but inertial sensors such as accelerometers and rate gyros, do not sense the turbulence directly.

IV.2 - Simulation results

This section presents some representative results using this failure detection procedure with a lot of sensor failure configurations (such as abnormal offsets, drifts, saturations, jumps ...) in conjunction with the complete non linear model.

In calm air, the outputs of this non-linear simulation fit with flight data, the N-262 aircraft and the model being subjected to the same standard inputs on the three axes.

The simulated aircraft is at various design flight conditions (speed and altitude) and excited by a three dimensional turbulence with Dryden spectral form. The simulated sensor outputs were corrupted by two kinds of errors.

- an unfailed bias
- an electric gaussian noise

TABLE 1

sensor	measurement	unfailed bias	R M S (standard deviation)
Inertial	ground speed V	0,5 m/s	0.25 m/s
	pitch angle θ	0.2°	0,06°
	vertical acceleration γ_{zo}	0.05 m/s ²	0.003 m/s ²
air data	angle of attack α	0.5°	0.06°
	air speed U	1 m/s	0.3 m/s
accelerometer	longitudinal γ_x	0.05 m/s ²	0.003 m/s ²
	normal γ_z	0.05 m/s ²	0.003 m/s ²
rate gyro	pitch rate q	0.05 °/s	0.03 °/s

The detection threshold ϵ_d is $3\sqrt{2}$ times the magnitude of the associated sensor RMS, and the rejection threshold three times the RMS.

The choice of detection and rejection thresholds is a delicate problem because the real sensor noises are not white and Gaussian; and this determination affects the false alarm probability.

The six-degree-of-freedom simulation is used to determine system performance under failure. The simulation allows inclusion of lateral and longitudinal autopilot in order to see the effect of a failure feedback.

A comparison of the simulated and estimated outputs is shown in Figure 5. In this case, the pitch and roll attitude holds are switch on and simultaneous failures (drift) are introduced at time $t = 2$ s on unlike sensor outputs. The behaviour of the predicted and estimated values is shown in figure 6 for the angle of attack measurements.

As far as the simulation is concerned, we obtain good results and rates of false alarm and wrong isolation are very low.

The form of the analytical redundancy formulation given in this paper, possess computational benefits relative to other approaches.

The level of redundancy available on the N-262 aircraft is high enough and leads to first order relations.

V. CONCLUSION

On the simulation, the results of the new procedure are as good as those obtained with the previous one, avoiding the main inconvenients discussed in the text. The elimination of the unknown inputs allows us to ignore not only the turbulence but the non measured controls such as thrust. We do not need a statistical model of these perturbations, but we need the knowledge of their effects on the aircraft dynamic. This formulation is well-adapted to a parallel implementation increasing the overall safety.

Before the final step : implementation on an on-board computer and in flight test; we are analyzing the behaviour of the estimates with real flight data in order to see the sensibility of the flight condition and the impact of all the perturbations.

This new approach points out and leads to a systematical research of all the deterministic analytical redundancies. The multi-criterion choice allows to take into account practical constraints and could include the sensor failure probabilities when they are known.

This approach leads to a generalisation of the use of skewed instrument in inertial systems [15]. On one hand, the automatic control needs independent state measurements. On the other hand, the failure detection would need coupled measurements, and this technique can be a guide for the definition of well-adapted sensor.

THANKS :

The research has been carried out on contracts from STAé (Service Technique Aéronautique) and DRET (Direction des Recherches et Etudes Techniques).

REFERENCES :

- 1 J.C. WILCOX :
Retroactive failure correction for strapdown redundant inertial instruments :
J. Spacecraft Vol. 12, n°6, June 75
- 2 J.E. POTTER, J.C. DECKERT :
Minimax failure detection and identification in redundant gyro and accelerometer systems - Journal of Spacecraft Vol. 10, n°4, April 1973
- 3 T.B. CUNNINGHAM, R.D. POYNEER :
Sensor failure detection using analytical redundancy - Joint Automatic Control Conference 1977, Proceedings 278-287
- 4 J.C. DECKERT, M.N. DESAI, J.J. DEYST, A.S. WILLSKY :
Reliable dual redundant sensor failure detection and identification for the NASA F8 DFBW aircraft - Rapport NASA, CR 2944, February 1978
- 5 R.N. CLARK, C.J. MASSELIEZ, J.W. BURROWS :
A fonctionnaly redundant altimeter - IEEE Trans. Aerospace, July 1976
- 6 R.N. CLARK :
Instrument fault detection - IEEE Trans. Aerospace, Vol. AES 14, n°3, May 1978
- 7 M. LABARRERE, B. GIMONET, A. BUCHARLES :
Etude de la détection de pannes de capteurs à bord d'un aéronef - Rapport de contrat STAé/DERA-CERT - January 1976
- 8 M. LABARRERE, B. GIMONET, A. BUCHARLES :
An estimation technique applied to failure detection - Congrès IFAC - Helsinki June 1978
- 9 R.C. MONTGOMERY, D.B. BRICE :
Management of analytical redundancy in digital flight control systems for aircraft - AIAA Mechanic & control of flight conf., Anakeim, Aug.5-9, 1974
- 10 T.T. CHIEN :
An adaptative technique for a redundant sensor navigation systems- AIAA Guidance and Control Conf., Stanford, Aug - 16/16, 1976
- 11 A.S. WILLSKY :
A survey of design methods for failure detection in dynamic systems - Automatica. Vol. 12, pp. 601-611, 1976
- 12 M. LABARRERE, J.P. KRIEF, B. GIMONET :
Le filtrage et ses applications - Editions CEPADUES, 1978
- 13 B. ETKIN :
Dynamics of atmospheric flight - Wiley, 72
- 14 M. LABARRERE, M. PIRCHER, S. TRABULSI :
Détection de pannes de capteurs par utilisation de la redondance analytique - Rapport CERT/DERA, Aug. 78
- 15 J.E. POTTER, M.C. SUMAN : Thresholdless redundancy management with arrays of Skewed Instruments - AGARD, AG. 224-77

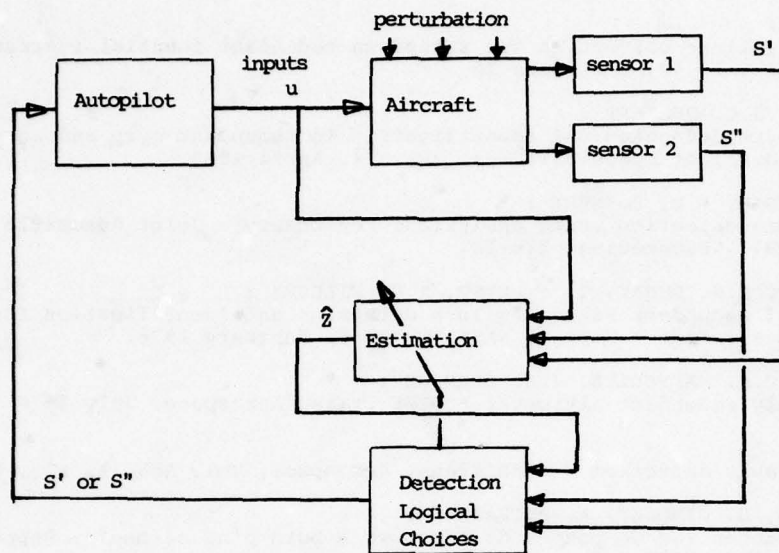


Figure 1 : Principle of the detection

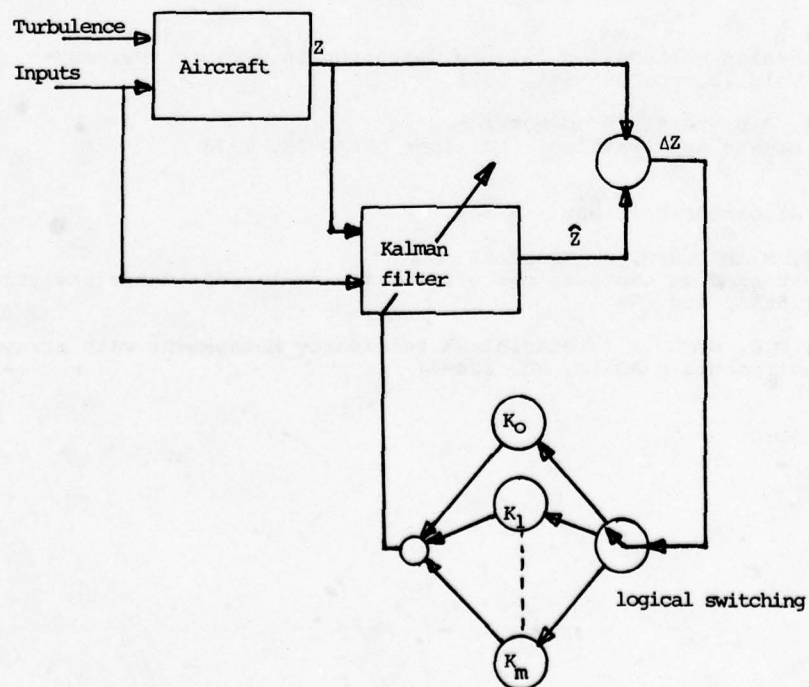
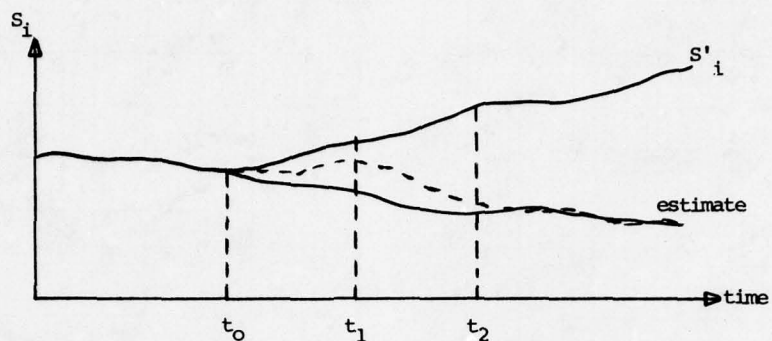


Figure 2 : The estimator



t_0 = occurrence of a failure on the first sensor

t_1 = failure detected : the two doubtful measurements are ignored

t_2 = isolation of the failed sensor

Figure 3 : Logical procedure of choice

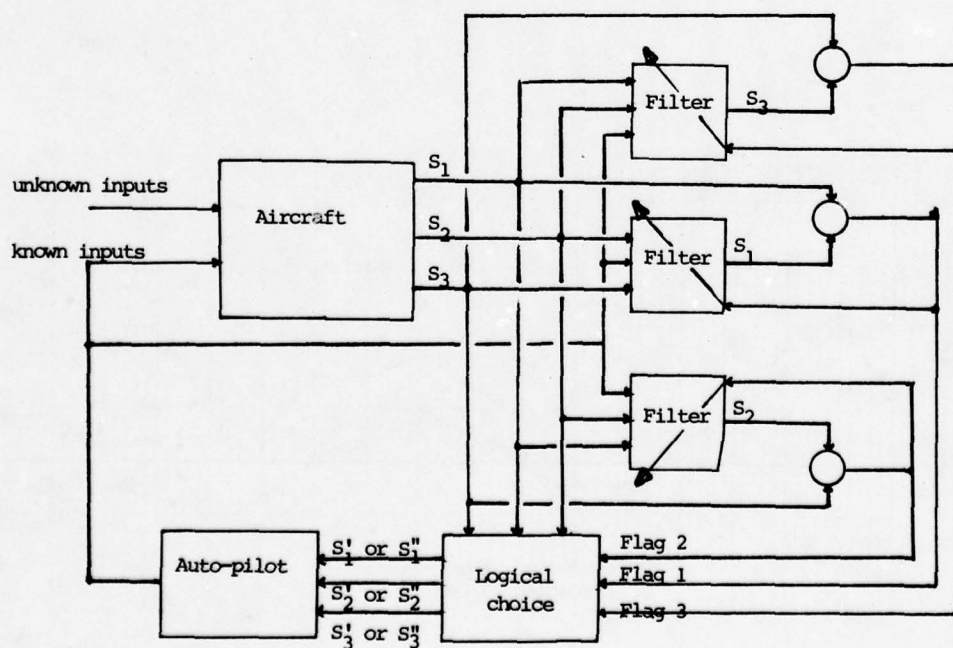


Figure 4 : Principle of the adapted analytical redundancy procedure shown on an example with three outputs.

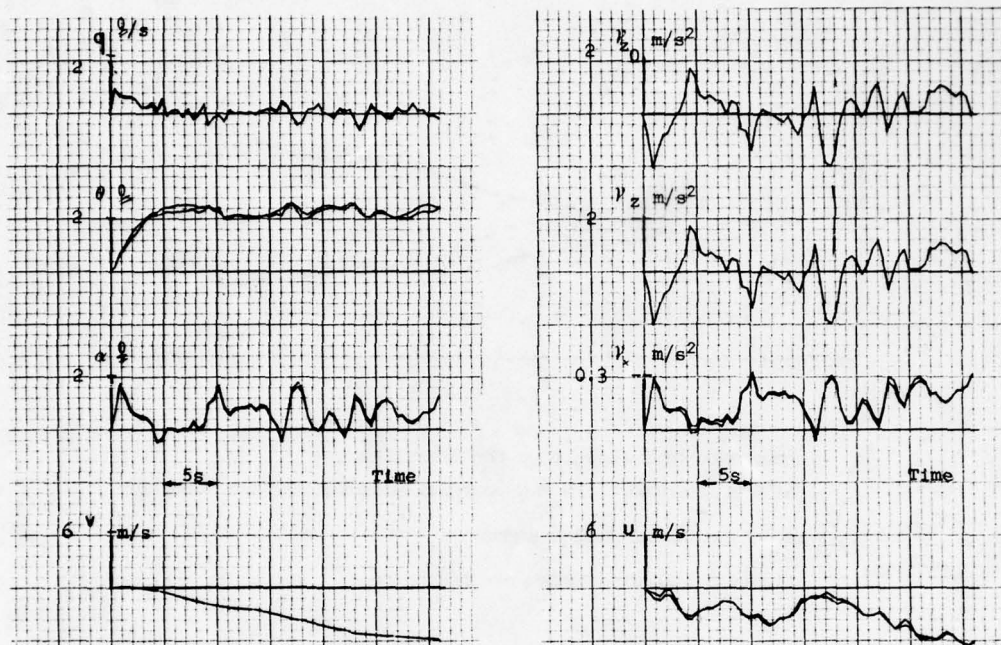


Figure 5 : Measured and estimated outputs with 3-D wind

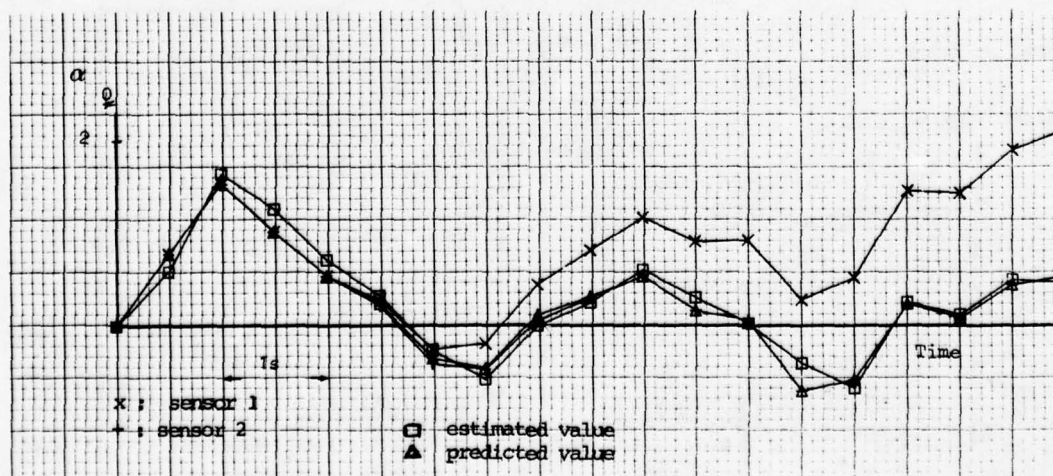


Figure 6 : Alpha vane failure

RECENT ADVANCES IN FIBRE OPTICS FOR HIGH INTEGRITY DIGITAL CONTROL SYSTEMS

by
R P G Collinson
Manager
Flight Automation Research Laboratory
Marconi Avionics Limited
Airport Works
Rochester
Kent ME1 2XX
England

SUMMARY

The use of fibre optics in active control systems offers high data rates with immunity from such things as electro-magnetic interference and lightning strikes.

In addition it gives complete electrical isolation, thereby eliminating the possibility of propagating electrical faults between interconnected units.

The paper discusses methods for using fibre optic cables for interconnecting the elements of an active control system and the advantages and disadvantages are discussed.

The major factors in the use of fibre optics are practical ones - connectors - terminations - ruggedness and environmental capability of cables.

The paper describes the techniques which have been developed to make a fibre bundle cable link a practical solution which can be exploited without risk.

The use of multi-access optical highways, particularly for interfacing other systems with the flight control system, (eg Air Data and IN systems) is reviewed and principles of the candidate networks outlined.

Finally, a new concept for a fibre optic multi-access network is presented which is fully compatible with the new MIL STD 1553B data transmission specification.

1. INTRODUCTION

Active Control Systems (or "Fly-by-Wire") offer major advantages in terms of performance and mission effectiveness.

Figure 1 illustrates a typical future advanced combat aircraft with Active Controls. The control surfaces are commanded by computer signals derived from the control stick demand and the various motion sensor feedback inputs so as to achieve the optimum response.

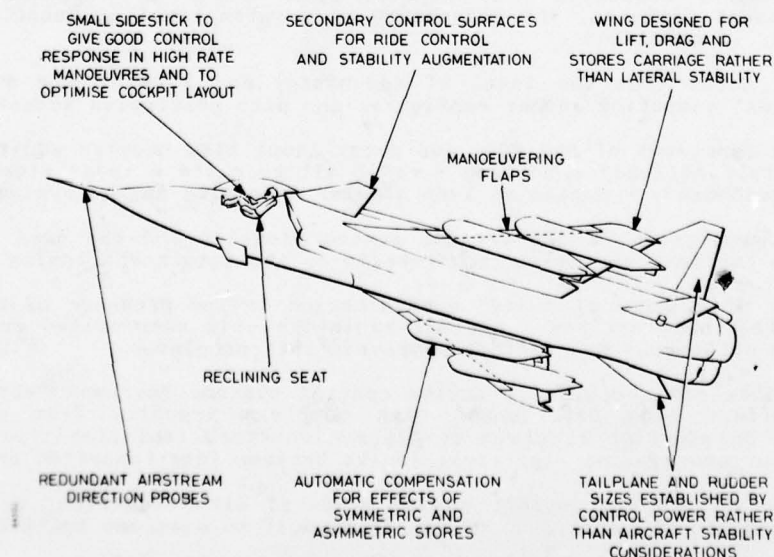


Figure 1 Typical Future Combat Aircraft with Active Controls

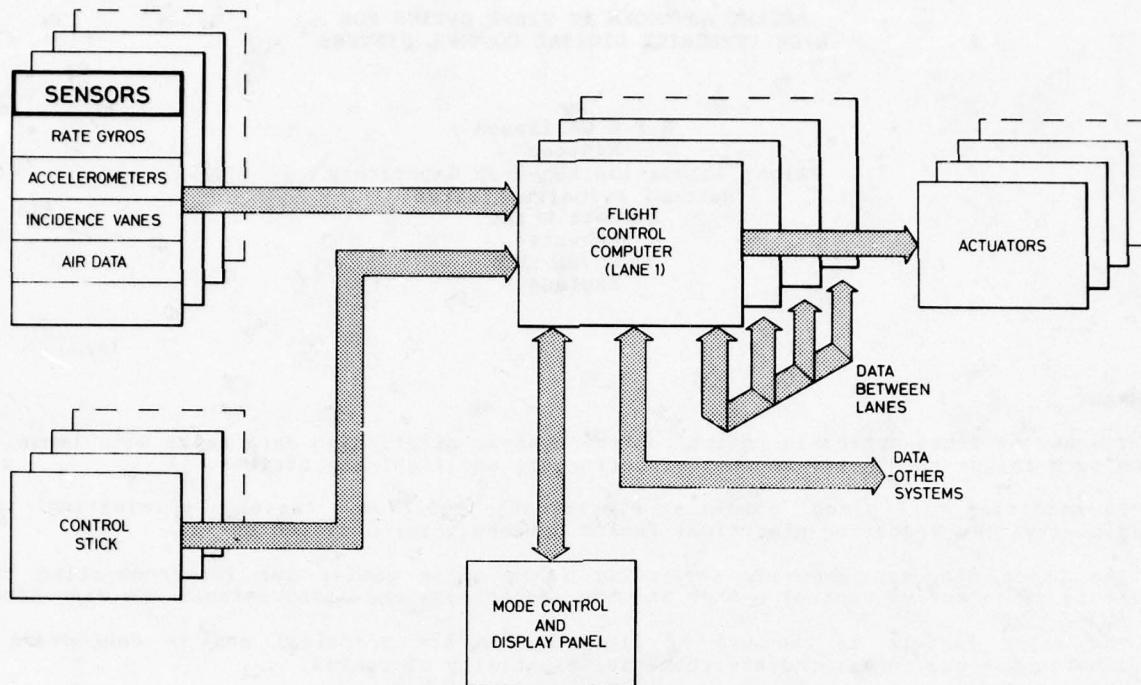


Figure 2 Data Flow in Typical Multi-Lane FCS

The block diagram of a typical multi-lane flight control system is shown in Figure 2.

The basic sensors comprising Pitch, Roll, Yaw Rate Gyroscopes; Normal, Lateral, Fore and Aft Accelerometers; Angle of Attack, Side-slip, Incidence Vanes; Altitude, Indicated Airspeed, etc are usually located some distance apart from each other. (The rate gyros and accelerometers may be at different fuselage stations because of the effects of body flexure; the air data sensors may be located near the Pitot-Static Probes to minimise pneumatic lags etc).

The multiplicity of control surfaces and their widely separated locations also contribute a large number of links for transmission of control signals to the actuator control valves together with position feedback signals from the actuators.

Redundant Sensors, Computers and Actuators are needed to overcome the effect of individual component failures, and redundancy at triplex level at least, and frequently quadruplex level, is specified.

It should be noted that the level of redundancy may vary in the sub-systems - eg "monitored Triplex" computing/sensor configurations with quadruplex actuation.

The autopilot functions of the FCS, (or outer loop) also require additional data such as - pitch and roll attitude - heading - radio altitude etc - these signals may be at a lower level of redundancy - duplex or even simplex depending on the system requirements.

The physical separation of the various system elements and the need to interconnect them thus places the most stringent requirements on the data transmission system.

The integrity of electrical "wire" communication in the presence of electro-magnetic interference - lightning strikes - nuclear radiation - is compromised and the extensive use of composite structures may further aggravate this problem.

The use of fibre optic cables in active control systems (perhaps "FLY-BY-LIGHT" is a better name) offers high data rates with complete immunity from electro-magnetic interference. In addition it gives complete electrical isolation thereby eliminating the possibility of propagating electrical faults between interconnected units.

There is also a potential weight saving compared with electrical data transmission systems, which may require shielding in copper conduit to overcome EMC problems.

2. BASIC REQUIREMENTS

2.1 Types of Link

Three basic types of link are required in an active flight control system and these are described briefly below, together with the applicability of fibre optics.

2.1.1 "In Lane" Links

These are the types of link required to couple the sensors to the FCS computers.

The fibre optic link (transmitter - connectors - cable - connector combination) should be of higher reliability than the basic sensor so that the overall reliability of the series combination is virtually that of the sensor alone. However, the complete immunity of the cable to electro-magnetic interference including lightning strikes and nuclear radiation (providing suitable fibres are used) greatly increases the overall system integrity. (It should be noted that the electronics conversion circuitry for the light pulse transmission and detection must be properly shielded and screened in the Sensor and Computer boxes respectively - this does not present a fundamental problem and is a question of careful design).

2.1.2 "Cross Lane" Links

These are the links used to cross-feed data between the computing lanes.

Data from one lane is required to be fed into the other computing lanes for the following reasons:

- (i) Faults are detected by cross comparison of the line data and by voting.
- (ii) Sensor signal consolidation to equalise the signals to prevent a "walkaway" with the time-integral terms used in the control laws due to individual sensor errors - offsets etc.

It should be noted that the individual lane computers require their computing iteration periods to be time-synchronised to avoid data staleness effects - this is generally accomplished by software in the computer executive programs using a synchronisation word in the data cross-feed.

It is clear that the cross-lane links require the highest possible integrity and the possibility of a common mode failure propagating between the lanes must be a negligibly low probability occurrence.

An electrical link even using electro-optical isolators may not meet the extreme integrity required.

A fibre optic link, however, can be seen to have the required intrinsic integrity giving complete electrical isolation with no means of corrupting the data along the link no matter what the source of electro-magnetic interference.

The cross-lane link has in fact provided one of the first flight control applications for fibre optics, and the digital flight control computing system produced by Marconi Avionics for the Boeing YC-14 uses such links. The system has now been operating satisfactorily for several hundred flying hours - the fibre optic links have shown no problems whatsoever.

2.1.3 "External" Links

These are the links required to couple data from other sub-systems such as the IN system - Air Data etc for the outer loop auto-pilot functions.

The data from these sub-systems is now generally available on a multiplexed electrical highway as specified in MIL STD 1553.

A means of "importing/exporting" data between the 1553 MUX BUS and FCS, which does not compromise the latter's integrity, is thus required. This access would also be advantageous in some reversionary modes.

A "fibre optic stub" which can interface directly with a 1553 electrical MUX BUS solves this problem and such a device is described in more detail in Section 8.

2.2 Data Rates

The data rates required in each of the paths shown in Figure 2 vary widely. In general, the in-lane links to sensors, actuators and control stick are each of low bandwidth and have traditionally been analogue. The bandwidth generally lies in the region of 30 Hz to 200 Hz, with a resolution of typically 10 to 12 binary bits, but sometimes as high as 16, for each motion sensor.

This gives average data rates from 300 to 3200 bits per second for each channel but a maximum of 1500 bits per second is more typical. Using instantaneous serial bit rates as low as this average value leads to delays in data transmission of one iteration period.

This is normally totally unacceptable in closed loop systems due to the destabilising effect this transport lag has on the loop stability. Actual data rates needed are at least an order higher than the average figures indicated above.

The data rate required for cross-lane links is very much higher.

As already mentioned, data is cross-fed for:

- (i) Failure monitoring
- (ii) Equalisation of the system states (integrators) to prevent noise, offsets and other minor errors accumulating
- (iii) Voting or consolidation of the sensor input data.

To achieve this for a typical 3-axis flight control system, some 64 parameters may need to be cross-fed at rates of 200 times per sec. With the conventional computing word length of 16 bits, this gives a total of 200k data bits per second between each pair of systems. Additional addressing and control data may also be necessary.

To minimise interconnections between lanes, it is usual to multiplex all the data into one link and very careful design is required to implement these cross-lane links.

The average data rates required in the links to external systems are usually similar to those of the in-lane sensor links. The actual data rates used tend to be very high as modern system design is leading to common multiplexed data transmission buses where data rates can be 1M bit per second as in MIL STD 1553.

3. OVERVIEW OF FIBRE OPTIC DATA TRANSMISSION TECHNOLOGY

A brief review of the technology is appropriate at this stage and this is set out below.

A fibre optic link has 4 main operational parts:

- 1. The fibre optic cable
- 2. Connectors for joining cables
- 3. The optical transmitter
- 4. The optical receiver

3.1 Fibre Optic Cable

A fibre optic cable consists of the relatively fragile fibre, or fibres, covered by a strong sheath to give mechanical protection.

The fibre optic itself consists of a central transparent core of high refractive index surrounded by a transparent cladding of lower refractive index. Operation is best understood by considering the light to be constrained to travel along the core by reflection or bending of the light rays at the transition between the different refractive indices.

The transition between the two indices can be abrupt, termed a 'step index' fibre, or can be gradual, to give a 'graded index' fibre. The latter gives a very high bandwidth even for long lengths but is very expensive. The step index fibre is cheaper and has adequate performance even for the most demanding avionic application.

Glass is the most widely used material, but plastic fibre optic is also used. This is very cheap but is not suitable for the harsh avionic environment. Silica can be made in very pure form and can give very low attenuation as well as very high resistance to radiation damage. Suitable cladding materials are difficult to find however and so far only certain plastics are suitable. They tend to be rather soft, however, and practical avionic handling and termination techniques have yet to be found.

The diameter of the active core of the fibre can be very small, only a few wavelengths of light in the case of optical waveguide, or can be several hundred microns diameter in the case of the recent 'fat' fibres. This parameter again affects the bandwidth but more importantly affects the mechanical tolerances needed to feed light into the fibres. A convenient way of effectively increasing the active diameter is to use many small fibres in a bundle and much practical work has been done on this arrangement (see later).

Another important practical parameter which affects the light gathering power of a fibre is its numerical aperture. In simple terms this can be regarded as a measure of the size of entrance cone from which light rays will be 'captured' by the fibre. Light rays outside of this cone are severely attenuated or escape through the cladding.

A great deal of investment has been expended in developing a wide range of types of fibre to meet the very high performance telecommunications applications. Except for radiation resistance, however, actual fibre performance requirements are less onerous in

the case of avionic applications, but meeting the practical installation and maintenance requirements is very demanding. This affects both the fibres themselves and the protective sheathing and this means that fibres must be chosen for their convenience of termination and handling. To date, only large active diameters and large numerical apertures successfully meet these requirements.

The sheathing material must give adequate mechanical protection to the relatively fragile fibre during installation handling and must resist cable clamping forces. Strength members can be used when heavy longitudinal stress is expected, but this is not normally necessary for aircraft applications.

Full environmental testing and many actual aircraft installations have illustrated that fibre optic cable can adequately withstand the aircraft environment.

3.2 Connectors

The structure of the cable affects the ease with which connectors can be implemented. Fibre bundle cable has a large optically-active area and hence slight misalignments due to clearance in the connector shells can be tolerated. This is also true of the 'fat' single fibres of several hundred micron diameter, but this aspect severely restricts the use of thinner single fibres on purely practical grounds.

3.3 Optical Transmitters

The main characteristics of a fibre optic transmitter are the ability to launch as much light as possible into the core of the fibre, and have an adequately high bandwidth. This means that a very intense light source with a small active area is needed, and lasers and light emitting diodes (LED) are suitable. Semiconductor lasers are more suitable but LEDs prove to have a higher reliability at the present state of the technology for these applications.

LEDs and semiconductor lasers are derived from similar technologies and any improvement in one usually reflects into the other so that it seems likely that LEDs may stay ahead in the race for some time.

The Burrus diode type of LED is frequently used. These diodes have a small diameter well, etched into the surface, which gives a very high current over a very small area. This gives a very small, very intense light source which can be easily coupled into a fibre optic.

Currently, such a LED taking 300mA pulses can launch one milliwatt of optical power into 0.5 numerical aperture 100 micron diameter fibres. A data rate of several tens of Mega-bits per second throughout the -55°C to +125°C aircraft temperature specification can be achieved.

3.4 Optical Receivers

Optical receiver diodes are limited by the minimum detectable optical power. This is set by the fundamental sensitivity of the device and its inherent electrical noise together with that of the associated amplifier circuitry.

Small area PIN diodes are easiest to use and typically can reliably detect down to 0.5µW over the full military temperature range.

Avalanche diodes have better sensitivity but they need a low-noise voltage supply in the region of 100 volts, which must vary with the diode temperature. The circuitry is therefore more complex and costly than for PIN diodes which have adequate performance for most applications.

4. THE "A to B" FIBRE OPTIC LINK

The basic link to meet the majority of avionic data transmission requirements is the simple "A to B" link.

It will be shown in later sections that a multi-access fibre optic network can also be implemented with these basic links.

Development of the components for a basic link to meet the full military avionic environment was initiated in the UK under a Ministry of Defence (PE) contract awarded to Marconi Avionics Ltd over five years ago. The link system has been given the name MINILINKS and its salient features are set out below.

4.1 "MINILINKS"

The main objective of MINILINKS was to develop a complete set of military standard fibre optic components compatible with existing avionic system design and with airframe installation. An important feature of this contract was that the development of each component was carried out in conjunction with a relevant component manufacturer. This was to ensure that a supply of components was available after the conclusion of the programme.

System Specification

During the initial stages of MINILINKS a survey was carried out, the objective of which was to formulate system specifications which would fulfil the majority of first generation requirements.

The survey identified the following areas of development:

- PCB mounted terminals
- Avionic fibre optic connectors
- Avionic fibre bundle cable
- Shop Floor termination techniques

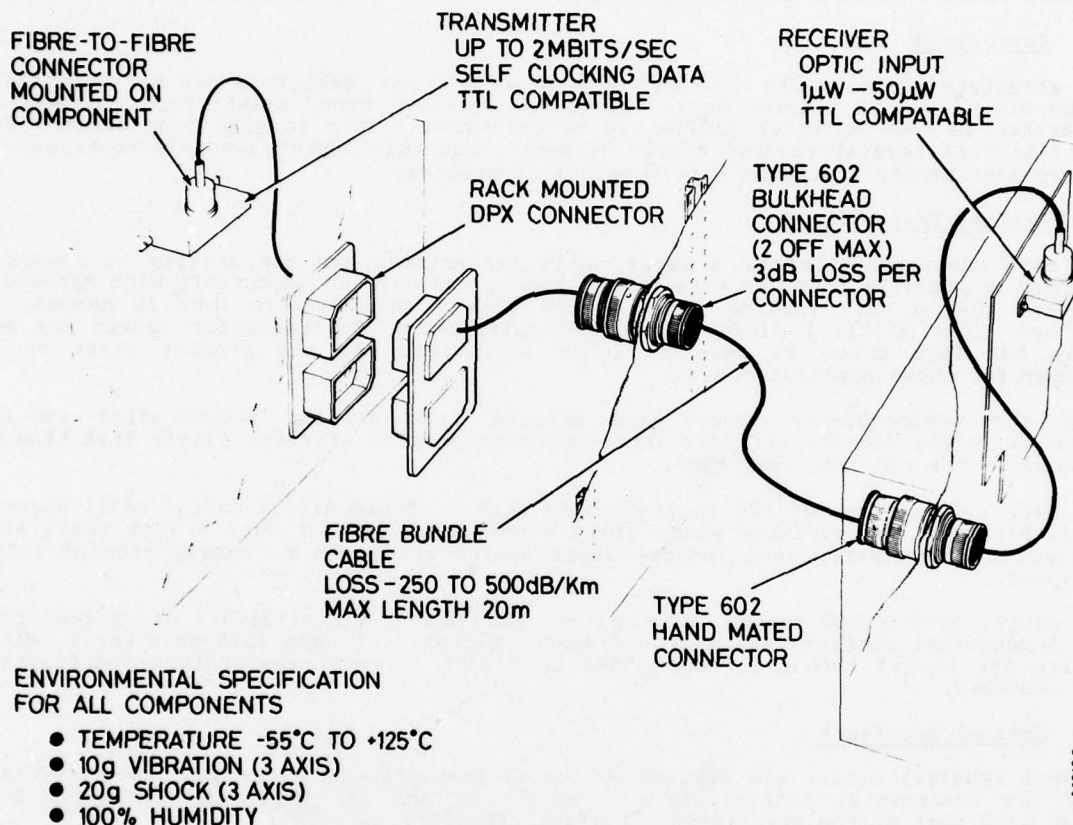


Figure 3 MINILINKS Fibre Optic Data Transmission System

Figure 3 shows a pictorial representation of the system specifications, the principle features of which are:

- Cable lengths up to 30 metres
- Number of connectors 2 to 6
- Data rates 200k bits/sec or 2M bits/sec clock and data.

It was decided to prove the suitability of the technology for avionic use by ensuring that the MINILINKS system would operate in an environment specified by the more rigorous sections of BS-3G 100 which can be summarised as follows:

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Humidity	up to 100%
Vibration	10g up to 2 kHz
Shock	20g maximum.

Fibre Optic Terminals

The MINILINKS system, being an A-B simplex link, required a transmitting (Tx) terminal and a receiving (Rx) terminal. The main features of the terminals were that they had to be a PCB mounted component of similar size to, and as easy to use as, conventional integrated circuits.

The operating temperature range combined with the inherent thermal characteristics of electro-optical devices required the development of appropriate circuitry.

A major design feature is the ability to cope with up to 6 in-line connectors which may be required for installation and servicing requirements.

The major source of attenuation is in these connectors (typically 3 dB) and taking the temperature extremes and the possible variation of from a minimum of 2 connectors to a maximum of 6, gives a dynamic range of 50 : 1 to cope with, in terms of attenuation.

Suitable miniature hybrid modules similar to those shown in the illustration in Figure 3 have been developed. (Later versions are designed for flat mounting on the PCB).

Fibre Optic Avionic Connectors

At the outset of the development programme it was decided to use existing avionic connectors. The primary reason for this decision was that the time and cost involved in the design and tooling for a new type of avionic connector was prohibitive for a new technology.

The two basic types of avionic connector are the rack-mounted or DPX connector and the hand-mated or Type 602 connector. The majority of design effort was concentrated on the termination technique and on ensuring that it could be embodied in a ferrule which was compatible with the connectors, rather than on modifying the connector.

Avionic Fibre Optic Cable

A fibre optic cable has two basic elements, the conducting medium and the protective sheathing. During the MINILINKS contract only two conducting mediums were available, these were a fibre optic bundle, or a small-diameter single fibre. The use of existing avionic connectors with their associated tolerances precluded the use of the fine single fibre, thus MINILINKS was based on fibre bundle technology.

Prior to MINILINKS, two basic sheathing designs were being marketed. The first consisted of a thin-walled PTFE tube which offered insufficient mechanical protection to the fibre bundle, whilst the alternative design was a sheathing which contained discrete strength members. This offered excellent mechanical protection, but the mechanical termination of a discrete strength member is not an existing 'shop floor' technique. Another disadvantage is that the non-destructive inspection of the cable after termination to ensure that when the cable is under tension the fibre bundle is not under stress, is virtually impossible.

The sheathing design developed during MINILINKS was of a dual-sheath construction, comprising two non-flammable plastics. The outer sheath consisted of Tefzel or Kynar. This is a hard plastic and allows the fibre optic ferrule to be mechanically terminated to the cable by means of a 'hex crimp'. The 'hex crimp' is an existing avionic technique used to terminate electrical cables. FEP was used to form the inner sheath; this soft plastic prevented the sheathing from collapsing on a sharp bend.

The conducting medium consisted of 96 fibres which formed a 780 μ m diameter bundle and has an attenuation of 400-500 dB/km.

Termination Technique

An entirely new termination technique to replace epoxy bonding was required to allow fibre optic cable to be terminated on the 'shop floor'. The technique developed has become known as the 'Hot Crimp'. The basic principle is that the end of the fibre bundle is contained in a glass bead and then pushed into a hot taper. The temperature of the taper is such that the glass bundle and bead soften. The movement into the taper is sufficient to produce the crimping effect illustrated in Figure 4.

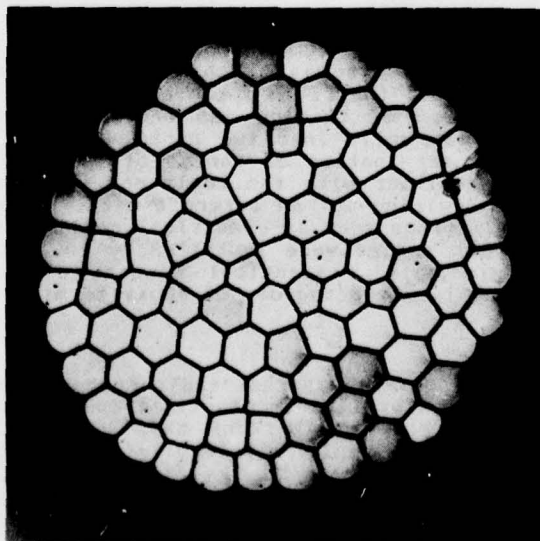


Figure 4 Hot Crimp Technique

The advantage of this technique is that the end face is 'all glass', and is thus much easier to polish. The crimping effect has also reduced the interstitial dead space between the fibres; this will result in a 30% reduction in connector loss.

Figure 5 shows how this technique has been incorporated into a Type 602 connector ferrule. The ferrule consists of three parts, a metal taper piece, a metal ram and the glass bead. Heat from a 'Hot Crimp' tool is applied to the taper piece. When the glass bead and bundle soften, a suitable force from the 'Hot Crimp' tool is applied to the glass bead via the metal ram and the fibre is compressed. It can be seen from the diagram that the process is controlled by the piece-parts and is complete when the gap, marked by a *, is closed. As the quality of the termination technique is controlled by the quality of the piece parts this technique is well suited to use on the shop floor.

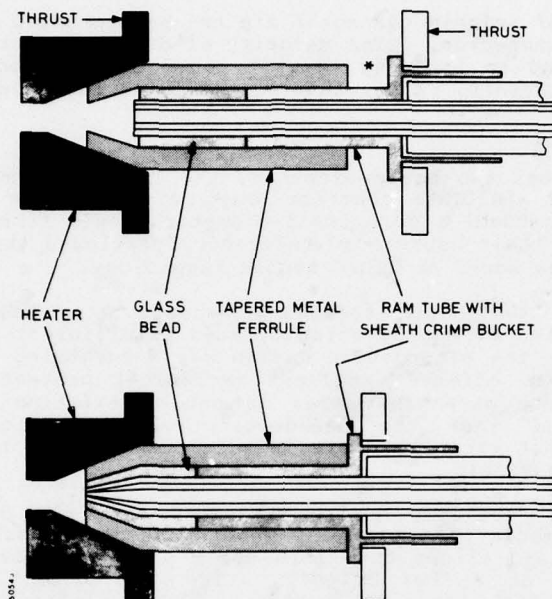


Figure 5 Hot Crimp Process

4.2 Aircraft Installation Trials

After the conclusion of the MINILINKS contract, the Ministry of Defence (PE) funded a joint contract between British Aerospace (Warton) and Marconi Avionics Ltd.

The main objective of this contract was to determine, from practical experience, the feasibility of fitting fibre optic components to production aircraft.

The contract commenced with Marconi Avionics Ltd carrying out a component survey. The purpose of the survey was to ensure that any components or techniques which had been developed independently of the MINILINKS contract were not excluded. A kit of parts consisting of epoxy and hot crimp termination techniques was purchased, together with avionic fibre optic cable and connectors.

The British Aerospace production team consisted of normal production wiremen, who were chosen on the basis of availability not skill, chargehands and electrical inspectors. Marconi Avionics Ltd demonstrated all the fibre optic tools, techniques and components to the production team. The period of tuition was one day. The installation work followed the demonstration and was split into two parts. The first part was a bench practice which consisted of the termination of cables of various lengths. This allowed the production team to become familiar with the tools and components. The second part involved the production team in 'wiring-up' an aircraft metal mock-up. The metal mock-up contained all the aircraft fittings such as electrical cabling, hydraulic pipes, ducts and LRU racks. The fibre optic cables were installed by the production team, with no immediate scientific supervision, in a identical manner to the existing electrical cables. One half of the mock-up involved the use of epoxy termination and the other half used hot crimp terminations.

The important feature of these trials was that their success was wholly attributed to normal skills of the production team applied to practical components and techniques. The trials have proved to British Aerospace (Warton) that the installation of correctly designed fibre optic components in small military aircraft does not constitute a technical risk.

5. MIL STD 1553B MUX BUS DATA TRANSMISSION

MIL STD 1553B MUX BUS has been adopted for airborne use in the US and is in the process of likely adoption for airborne use in the UK.

The wide and rapid acceptance of "1553" with all the benefits of agreed standard interfaces is resulting in a substantial investment in 1553 LSI components. The desirability of exploiting this investment and achieving compatibility with fibre optics is obvious.

A brief review of the MIL STD and the features which are relevant to any potential fibre optic multi-access network is given below.

The MIL STD covers all aspects required to implement an aircraft multiplexed data bus. The basic architecture consists of up to 31 terminals which are connected by an electrical T highway. One of the 31 terminals is a bus controller, which has total authority (command/response philosophy is used) over data flow on the bus, and the remainder are remote terminals.

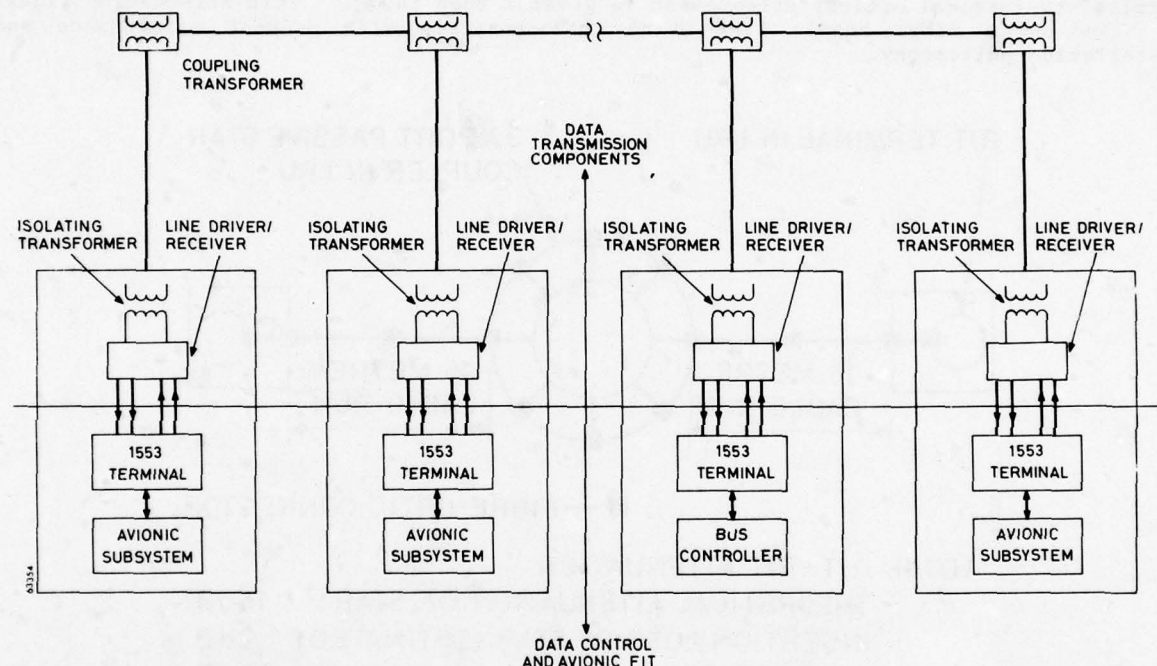


Figure 6 MIL STD 1553 MUX BUS

Figure 6 shows the basic format of a MIL STD 1553 system. The bus controller and remote terminals can be divided into two basic elements, the data transmission elements and the data control elements. The data control elements contain all the logic for controlling protocol, word format, error checking and subsystem interface. Any fibre optic multi-access network should only replace the data transmission elements and require no modification to the data control elements.

The data transmission elements are comprised of a common bidirectional electrical highway to which each terminal is connected by stub. Each highway/stub and stub/terminal connection is achieved by transformer coupling. Every terminal has a Line Driver/Receiver which converts four logic lines from the data control elements to a biphasic signal and vice-versa. The MIL STD has a data rate of 1M bit per second and the modulation technique is Manchester II Biphasic encoding.

Before a fibre optic multi-access network can replace the data transmission components of a MIL STD system and not require any modifications to the data control elements, it must be capable of achieving the following:

- (i) Relay one optical signal to up to 30 other terminals
- (ii) Half duplex data transmission
- (iii) Time delay due to the network must not impact on the inter-message gap specification*
- (iv) Must be capable of transmitting Manchester II Biphasic at 1M bit per second directly or using some form of intermediate modulation.

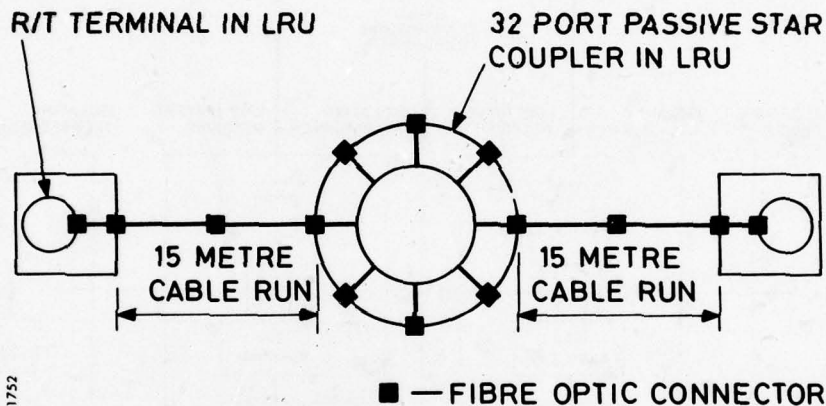
*Inter-message gap is defined as the time between the end of transmission from one

6. EXISTING FIBRE OPTIC MULTI-ACCESS NETWORKS

Many fibre optic network configurations have been formulated. The majority of them can be split into two basic types, the Central Distributor (or star), and the T Highway. Each of these configurations can be passive or regenerative in nature. A brief summary of these networks will be given in sufficient detail to justify the development of an alternative, FOREMAN. An important factor which must be taken into account is the maximum optical attenuation that can be tolerated in a military standard system. Currently, the maximum permissible attenuation is approximately 30 dB. This is based on the fact that the minimum detectable optical power at 125°C with a 1 in 10⁹ error rate is 1 μW, and the maximum optical power that can be obtained from a LED at 125°C is 1 mW.

6.1 Passive Central Distributor (Star)

This network is illustrated by Figure 7. It consists of 32 terminals which are linked by the central distributor. The central distributor is a passive optical component which accepts light from any optical terminal, evenly splits the light and distributes it to all the terminals in the network. The table below shows that the terminal-to-terminal optical attenuation is greater than 50 dB. This attenuation figure is based on a fibre bundle link which is compatible with present maintenance and installation philosophy.



TOTAL R/T-R/T ATTENUATION

THEORETICAL ATTENUATION OF STAR	15dB
INSERTION LOSS OF STAR (ESTIMATED)	2dB
CONNECTORS (8 OFF)	24dB
30 METRES OF CABLE (400dB/km)	12dB
	<u>≤53dB</u>

Figure 7 Avionic 32-Way Central Distributor

Total R/T - R/T Attenuation

	Bundle Cable (Typical)	Fat Single Fibre (Estimated)
Theoretical Attenuation of Star(1/32)	15 dB	15 dB
Insertion Loss of Star	2 dB	2 dB
Connector Losses (8-off)	24 dB	16 dB
Loss in 30 Metres of Cable	12 dB	-
	<u>53 dB</u>	<u>33 dB</u>

The configuration also has a number of inherent disadvantages. The physical distribution of the avionic LRUs in the airframe, combined with the requirement to connect to the central distributor, will result in long cable lengths. The central distributor component would constitute a common mode failure.

This configuration could be used in specific applications. The optical attenuation can be reduced to acceptable levels by reducing system and installation requirements. A network based on ten terminals connected by a one piece preformed loom, which would connect all the aircraft wiring and the central distributor element, would be feasible.

6.2 Active Central Distributor (Regenerative Star)

This network is similar in format to that discussed in Section 5.1. The main difference is that the central distributor element is active. This element would accept an optical signal from any of the terminals in the network, regenerate it and distribute it to the remaining terminals on the network. This network would result in an optical attenuation of less than 30 dB. This implementation overcomes the optical attenuation problem; however, the common mode failure component is now active and would result in a much lower network reliability.

6.3 Passive T Network

The configuration is shown in Figure 8, and would appear immediately attractive as it is identical in format to the electrical network specified by MIL STD 1553.

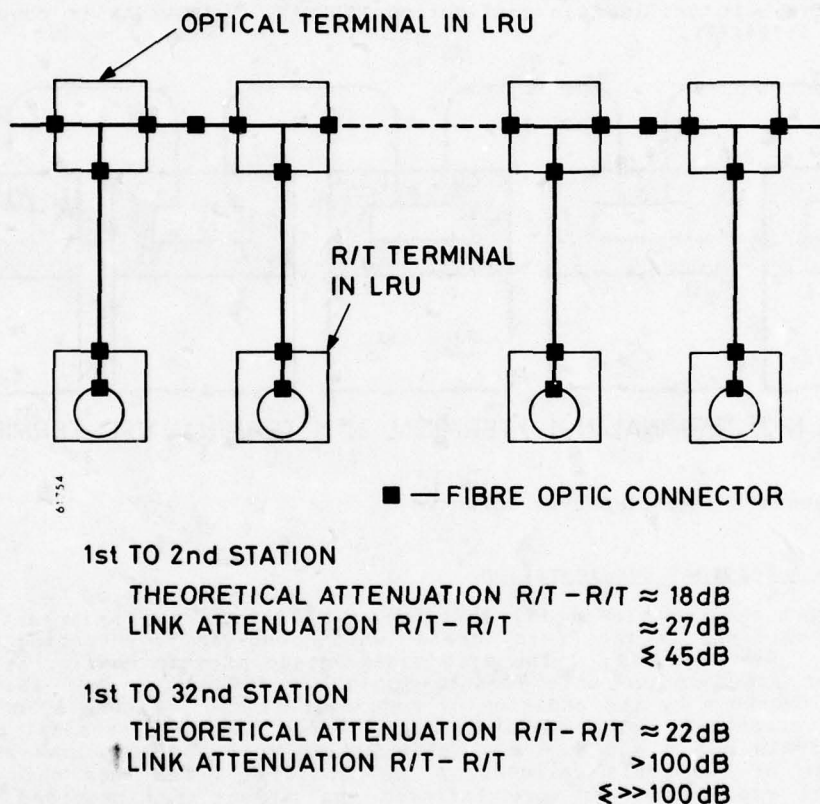


Figure 8 Avionic 32-Way Optical T Highway

The diagram shows that optical attenuation associated with this network is prohibitive. Fibre optic technology will require considerable development before this network would be practical unless considerable concessions are made in relation to the system and installation requirements.

6.4 Active T Network

The network discussed in Section 6.3 had a prohibitive optical attenuation caused by successive insertion losses due to the optical T pieces. This attenuation can be reduced to practical levels by implementing the network with active T pieces. The active T piece would be a regenerative unit which accepts optical signals, regenerates them and channels the regenerated signal through the appropriate ports. This network has the advantage of being practical in relation to today's technology. However, the inclusion of active T pieces introduces a number of disadvantages. This network would result in doubling the number of active LRUs which would normally be associated with the avionic fit of an aircraft. An aircraft electrical supply would have to be provided for each active T piece. This would greatly complicate the aircraft electrical supply system. Substituting active units for passive units will also reduce the reliability of

7. "FOREMAN"

After reviewing the fibre optic networks discussed in Section 6, it was felt that a new approach was required to allow a 1553-compatible fibre optic network to be implemented in the near future.

The new system developed by Marconi Avionics Ltd to implement a 1553-compatible fibre optic highway has been given the acronym "FOREMAN" standing for Fibre Optic Regenerative Multi-Access Network.

The initial development and proving of the FOREMAN concept was carried out on a private venture basis by Marconi Avionics Ltd. The system is now being supported by the Ministry of Defence (PE) who have awarded a contract to Marconi Avionics Ltd to produce a suitable demonstration system.

To reduce timescales it was decided to make maximum use of the existing military standard fibre optic hardware. The following section shows how the technology discussed in Section 4 has been utilised in configuring FOREMAN. (FOREMAN is covered by Patent Application No.45764/77).

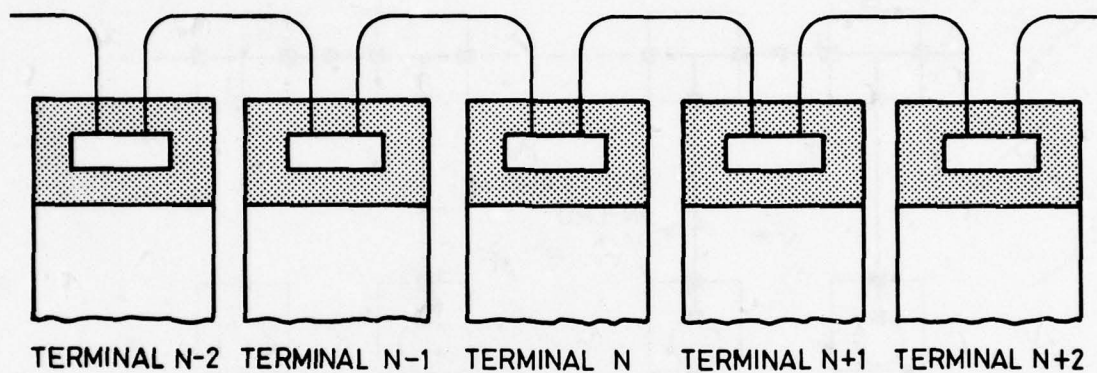


Figure 9 Regenerative Multi-Access Network

7.1 FOREMAN - External Configuration

The existing technology is wholly based on A-B links. The simplest multi-access network using A-B links is the 'daisy chain' which involves regenerating the signal at each terminal. See Figure 9. The main disadvantage of this configuration is that a single cable or regenerative unit failure would result in a system failure. This problem can be overcome by the addition of redundant links. Figure 10 shows how these links could be arranged. The configuration is such that any terminal N has optical access to terminals $N + 1$ and $N + 2$. This will give a minimum survivability of one terminal failure or two cable failures. It should be noted that this is a minimum failure survival capability, as more failures can be survived provided they are not adjacent.

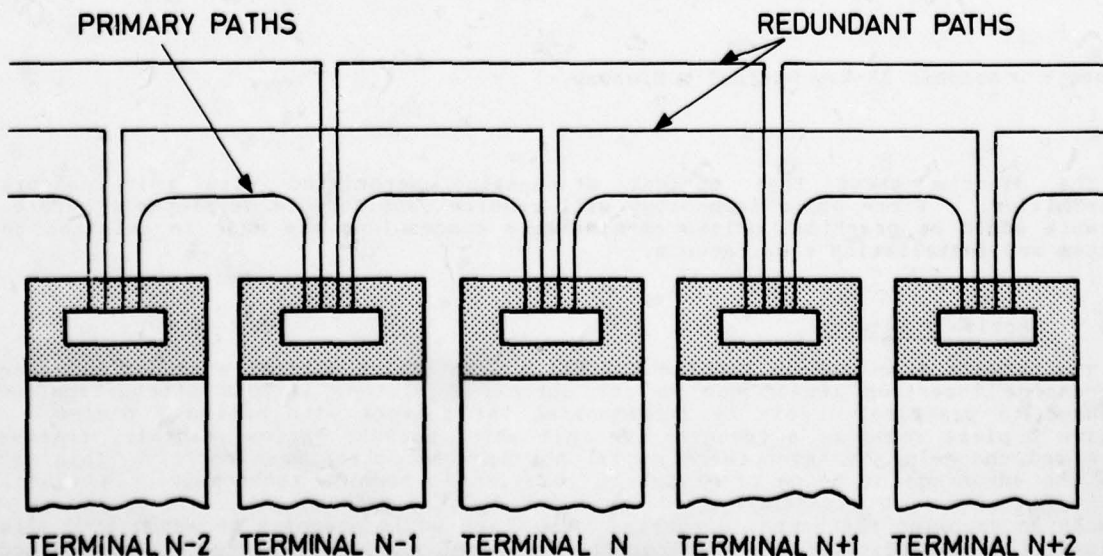


Figure 10 Regenerative Multi-Access Network with Redundant

7.2 FOREMAN - Basic Configurations

Figure 11 shows all the fibre optic components required to implement a FOREMAN system. The external components are those required to implement an A-B link, - a proven technology. The internal components consist of a 4-way fibre optic Y, which is a simple development of an existing 2-way fibre optic Y.

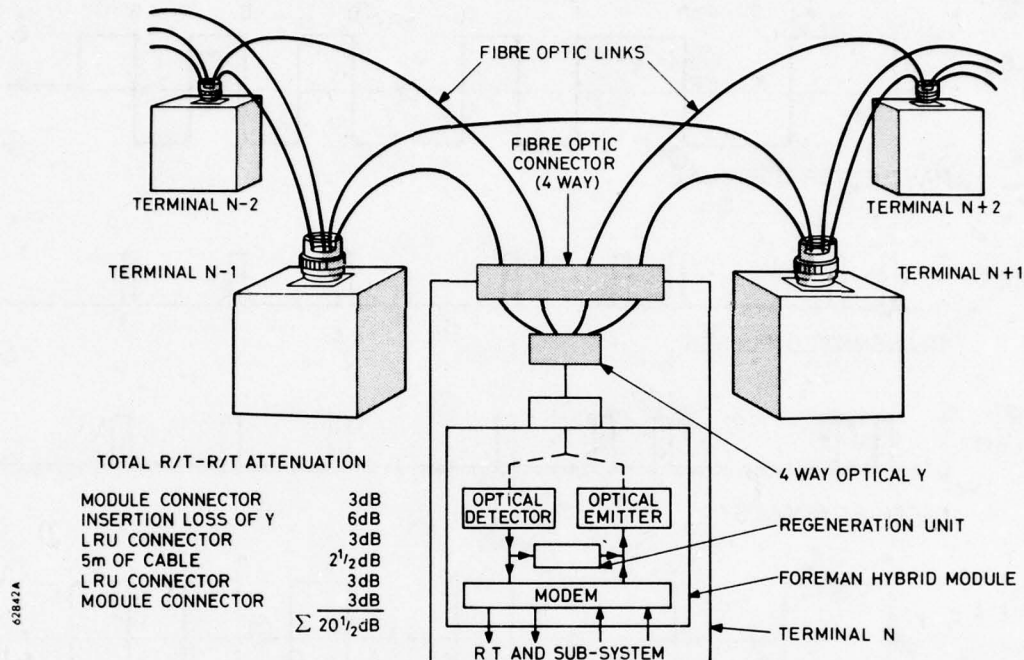


Figure 11 FOREMAN Fibre Optic Regenerative Multi-Access Network

This particular configuration of FOREMAN entails any one terminal having an optical path to the four adjacent terminals; thus, terminal N is connected by a fibre optic cable to terminals N + 1 and N + 2. It can be seen from the diagram that the four cables from terminals N + 1 and N + 2 enter terminal N by a 4-way connector. The four cables are then joined by a 4-way Y, the main stem of the Y being connected to a bi-directional terminal.

An important feature to note is that each LRU only has one active terminal and thus external redundancy is gained by the use of passive components.

7.3 FOREMAN - Principle of Operation

Another disadvantage associated with the networks involving successive regeneration is the accumulated propagation delays.

To attain compatibility with the MIL STD 1553 timing, short pulses of fixed width are transmitted around the network to reduce propagation delays to a minimum.

Figure 11 shows five terminals of a FOREMAN system and terminal N shows the basic internal elements. Consider the case of the optical emitter in terminal N-2 producing a light pulse of fixed width. This pulse would be relayed to terminals N, N-1, N-3 and N-4.

The following effect, which will be described in detail in relation to terminal N, occurs in terminals N-1, N-3 and N-4 simultaneously, (the latter two not being shown in the diagram). The pulse of fixed width from terminal N-2 enters terminal N via the 4-way connector. The 4-way Y directs this pulse to the bi-directional terminal. When the optical detector receives this pulse, a pulse of similar width is generated by the optical emitter. The 4-way Y will cause this regenerated pulse to be conveyed to terminals N + 1 and N + 2.

The pulse sent back to terminal N-2 has no effect, it being the originator; the pulse conveyed to terminal N-1 forms a redundant link, complementing the optical pulse transferred between terminals N-2 and N-1. The two remaining signals are conveyed to terminals N+1 and N+2. These signals will cause terminals N+1 and N+2 to be activated by terminal N in the same manner as terminals N-1 and N were activated by terminal N-2.

Thus, the regeneration process is reiterated as many times as is necessary for the original light pulse to be conveyed to all the terminals (31 maximum) in the network.

Figure 11 shows that the minimum FOREMAN configuration has an optical attenuation of 20.5 dB. This leaves a 9.5 dB margin on the 30 dB maximum transmitter-to-receiver attenuation allowable in a military standard system.

7.4 FOREMAN - Modulation Technique

The last major design problem was to formulate a suitable encoding technique which allows Manchester II coding to be represented by pulses of fixed width. Figure 12 shows the technique adopted. The Manchester II coded signal (trace 1) produced by the transmitting terminal is inspected by the appropriate FOREMAN terminal every 0.5µs.

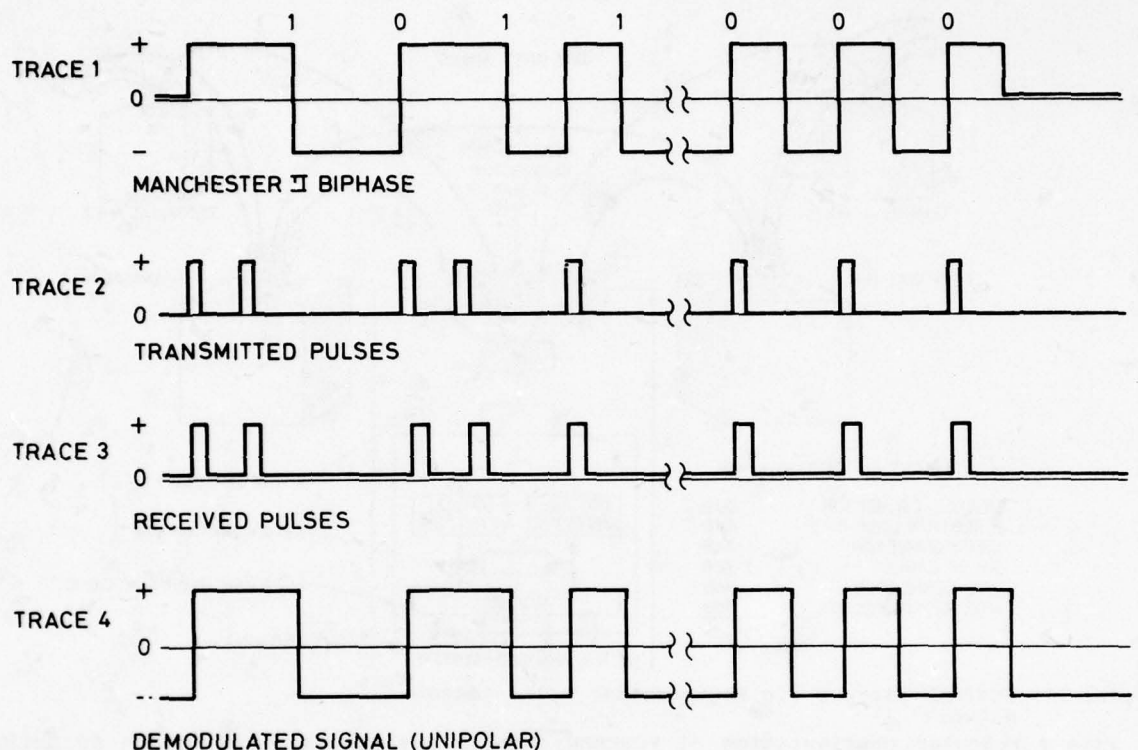


Figure 12 FOREMAN Optical Modulation Technique

If the level is high, a pulse of fixed width is transmitted along the network; alternatively, if the level is low, no pulse is transmitted. Thus, a stream of pulses of fixed width (trace 2) is formed and relayed to all the other FOREMAN terminals by the network. The receiving FOREMAN terminals demodulate this stream of pulses (trace 3) and reconstitute the original signal (trace 4).

7.5 FOREMAN - Interfacing Techniques

Complete compatibility with the MIL STD gives greater flexibility of implementation than would normally be expected from a fibre optic network.

The optimum point for interfacing with FOREMAN is between the driver/receiver unit and the Manchester Encoder Chip (or any future chip sets). This interface has four TTL lines and the Modem in the FOREMAN terminal has been designed to interface directly between these four lines (Manchester Biphase) and the optical regeneration circuit (train of fixed width pulses). This implementation involves replacement of the Driver/Receiver components, the isolating transformer and the electrical connector by the FOREMAN terminal, a fibre optic 4-way Y and a 4-part fibre optic connector.

Figure 13 illustrates an experimental development terminal implemented mainly with discrete components but with some optical hybrid modules.

8. THE "OPTICAL STUB"

The optical stub, which is shown in Figure 14, consists of an LRU with a simplified FOREMAN terminal which is connected by a fibre optic cable to the coupling LRU. The simplified FOREMAN terminal does not include the 4-way Y and does not require the regeneration circuit. Electrical coupling to a 1553 bus is usually achieved by a transformer which means that the coupling LRU is passive and requires no additional power supplies. To implement an 'optical stub', a simplified FOREMAN terminal and a transformer driver/receiver would also have to be incorporated into the coupling LRU, which would require an aircraft power supply.

As already mentioned, such an 'optical stub' can be used to couple flight-critical systems to a mission-critical MIL STD 1553 bus. This ensures total electrical isolation between flight-critical and mission-critical systems, but gives flight-critical systems access to mission-critical data.

AD-A076 146

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/G 17/7
ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQU--ETC(U)
AUG 79

UNCLASSIFIED AGARD-CP-272

NL

2 OF 4

ADA
076146



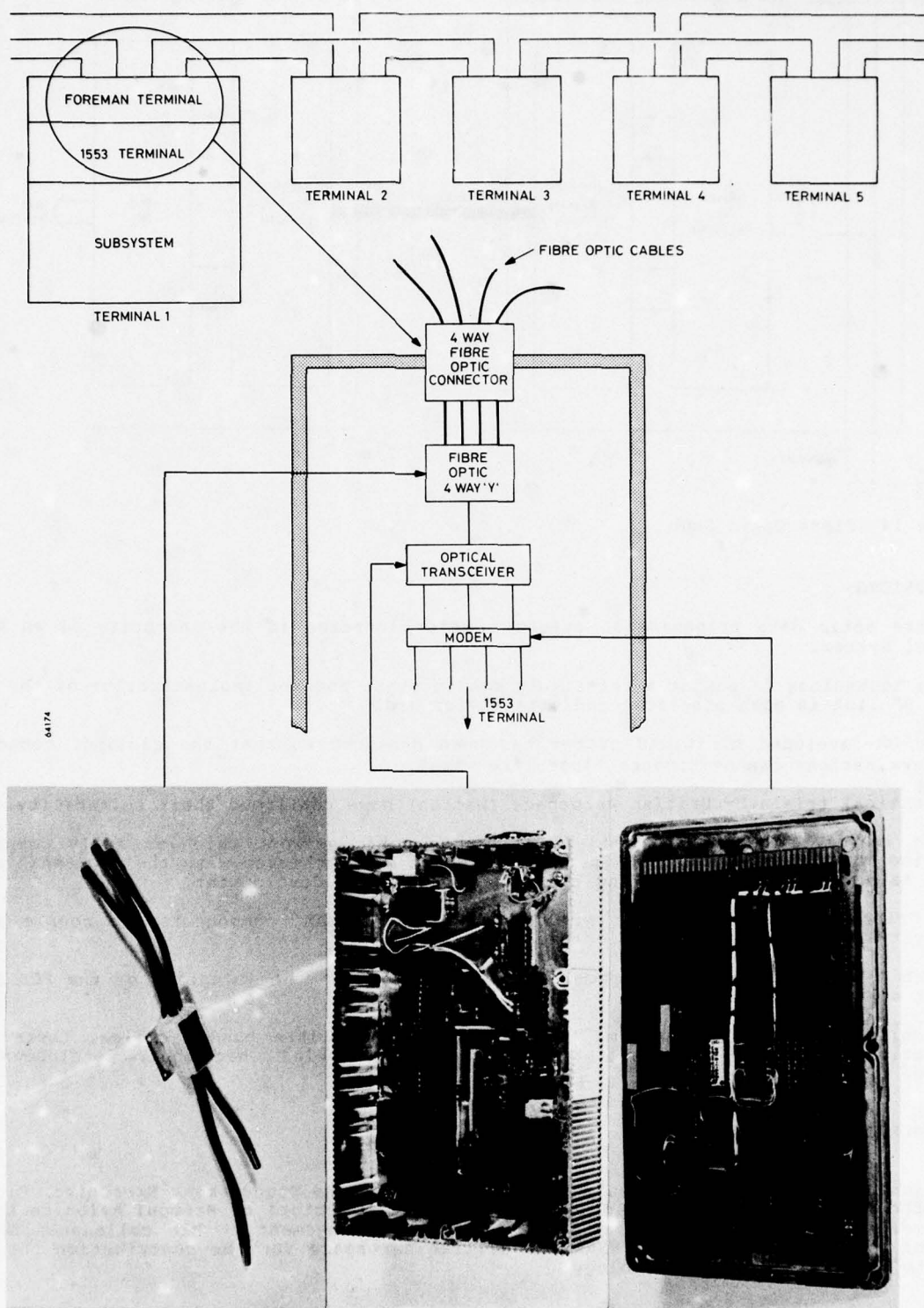


Figure 13 Experimental FOREMAN Terminal

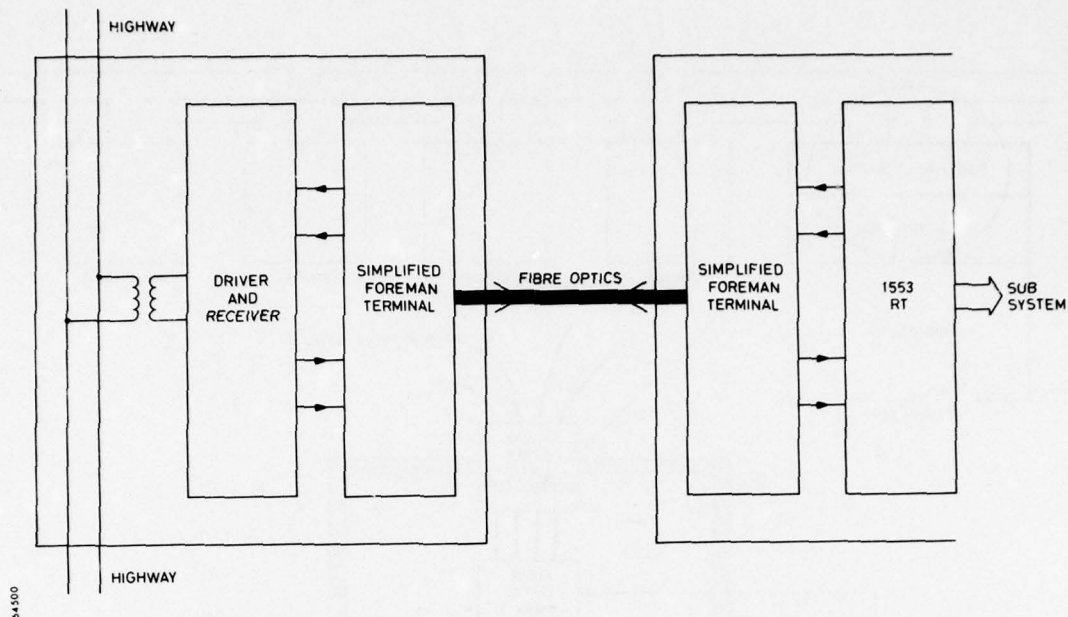


Figure 14 Fibre Optic Stub

CONCLUSIONS

Fibre optic data transmission offers a major increase in the integrity of an Active Control System.

The technology is now at a relatively mature state and the implementation of the basic "A to B" link is both practical and straightforward.

The UK-developed MINILINKS system has been designed so that the cabling, connectors and terminations can be treated "just like wire".

Practical trials by British Aerospace (Warton) have confirmed their suitability.

The implementation of a multi-access fibre optic network which is fully compatible with the MIL STD 1553B MUX BUS Specification has been achieved with the "FOREMAN" system which is able to exploit existing fibre optic component development.

A "fibre optic stub" can be produced, using "FOREMAN" components, to couple a high integrity FCS to the existing electrical MUX BUS mission system.

Such a system is entirely practical and ensures that the integrity of the FCS system is not compromised in any way.

Finally, although the systems described have used fibre bundle cables, there is no intrinsic limitation to substituting single fibre cables when their development has reached a comparable stage.

ACKNOWLEDGEMENTS

This work has been carried out with the support of the Procurement Executive, Ministry of Defence, and the author would like to thank the Directors of Marconi Avionics Ltd for permission to publish this paper and to pay acknowledgement to his colleagues both in Marconi Avionics and RAE, and RSRE and British Aerospace for the contribution they have made to this challenging technology.

REDUNDANCY MANAGEMENT CONSIDERATIONS
FOR A
CONTROL-CONFIGURED FIGHTER AIRCRAFT
TRIPLEX DIGITAL FLY-BY-WIRE FLIGHT CONTROL SYSTEM

By
John H. Watson*
William J. Yousey+
James M. Railey++

GENERAL DYNAMICS
Fort Worth Division
P.O. Box 748
Fort Worth
7X 76101, USA

ABSTRACT

In order to properly implement a digital fly-by-wire flight control system for control-configured fighter aircraft, it must be recognized that total shut down of the flight control computers (FLCCs), even for a fraction of a second, is paramount to the loss of the aircraft and its crew. To preclude the shut down of the FLCCs, redundant (parallel) processing is used in conjunction with redundancy management concepts.

The use of digital computing power offers the challenge of accomplishing this critical task with triplex redundant flight control computers without a degradation in overall system reliability. Using reliability requirements and goals as expressed in loss-of-control per flight hour, a digital flight control system (DFCS) architecture is evolved with specific emphasis placed on the input, processor and output subsystems.

The incorporation of an analog cross strapping of lower reliability sensors is shown to be an effective means of increasing system reliability by retaining sensor redundancy after a computer failure. A technique called control law reconfiguration is developed which insures system survival after a second like sensor failure. Computer contribution to loss-of-control is reduced by the addition of system monitors which increase the computer self-test confidence level. The resultant architecture is shown to have an inherent reliability which is relatively insensitive to the configuration of the actuator interface, thus allowing this interface to be designed based on hardware/software complexity tradeoffs.

This paper is based on work performed under Air Force Flight Dynamics Laboratory Contract No. F33615-77-C-3063 with additional information contained in Reference 1.

1. INTRODUCTION

Redundancy management is the process by which proper operation of the FLCC complex is monitored, faults are detected and isolated, and the remaining unfailed system elements are reconfigured. Traditionally, flight control systems have been designed to specific failure criteria, such as two-fail-operate, which inherently dictate system architecture, and then a reliability analysis is conducted to determine system loss rates, abort rates, and mean time between maintenance. The digital flight control system development task departs from this approach in that the loss-of-control and abort rates become the drivers in the design, have a large influence on the selected system architecture, and force reliability to be a major factor in the program. The sections on system reliability requirements and system development detail how an initial requirements assessment was performed to identify candidate architectures and how these architectures were evaluated and developed into a final configuration. A design goal was to meet the system loss-of-control and abort-rate requirements through the use of three digital computers and triplicated flight critical sensors. The section on control law reconfiguration describes how additional reliability can be achieved through the use of this technique after dual sensor failure. Various system monitoring techniques are discussed in the section on processor design. These tests can be used to improve the self test confidence level of the digital processor and related subsystems, thus increasing overall system reliability. The final section discusses the impact of various actuator interface designs on the total redundancy management system.

* Engineering Chief,
AFTI-F-16 Control of Flight Group
Professional Engineer, Associate Fellow AIAA
+ Engineering Specialist, Digital Flight Control Systems
++ Assistant Project Engineer, Flight Control Systems
Professional Engineer

2. SYSTEM RELIABILITY REQUIREMENTS

The development of the flight control system configuration was directed primarily by the reliability requirements and goals. To attain the loss-of-control requirement of less than one loss in 10^6 flight hours (a goal of less than 3 losses in 10^7 flight hours) and the abort requirement of less than one abort in 10^5 flight hours, the triplex digital system requires an architecture more complex than a simple tri-single-strand arrangement.

3. SYSTEM DEVELOPMENT

Before a system architecture can be developed and a reliability trade study accomplished, it is necessary to define the components of the system and assign piece part reliability figures to the subassemblies. The principle components required in a triplex flight control system are the flight critical sensors and controllers, the digital computers and the actuation system. The loss of control and abort requirements were stated to exclude the actuation system so this component will be neglected for the time being. Figure 1 presents the major subassemblies of a digital processor as used in a triplex control arrangement. A MIL-HDBK-217B analysis was conducted on an existing micro-computer and produced the mean time between failure (MTBF) data presented in Table 1 for the various subassemblies and the computer as a whole. The table also presents MTBF data for the primary sensors and controllers.

In order to simplify the reliability analysis, we will assume that any computer failure is a total computer failure (MTBF=1150 Hrs) and that the loss of two computers results in loss of control unless the second failure is successfully isolated by self test. The loss of three processors will always produce loss of control. We will also assume that all sensors and controllers are necessary for safe flight and thus the loss of two like sensors or controllers will result in loss of control. This implies that there is no self test capability of these functions in flight.

The simplest triplex configuration is shown in Figure 2 as System 1. This architecture assures that a single sensor and controller input from each redundant set is input to each computer. The computers calculate the appropriate output commands, exchange this information via the digital cross strap, perform cross channel monitoring and voting and then send the selected command to the actuators. The resultant loss-of-control probability equation has the following form:

$$P_{loc} = 3P_c^2(1-ST) + \sum_{i=1}^n 6P_cP_{si} + \sum_{i=1}^n 3P_{si}^2$$

where P_{loc} = probability of loss of control
 P_c = probability of computer failure
 P_{si} = probability of i_{th} sensor failure
 ST = self-test confidence level

The first term in the loss-of-control equation is the probability that two computers have failed and the computer self test has failed to isolate the failure. The second term accounts for the failure of a computer (which effectively causes the loss of hard-wired sensors) followed by a failure of a sensor in another branch. The third term accounts for dual-sensor failure cases. The equation is simplified in that it contains only computations of two failures and does not include the fact that the probability of the excluded parts of the system being good is not exactly unity. These exclusions are minor and have no bearing on the conclusions generated by the analysis. Assuming a 1.4 hour flight time, a 95% confidence level on computer self test, and the MTBF values of Table 1 it is determined that the second term (33.79×10^{-7}) is the major contributor to system loss followed by the first (2.22×10^{-7}) and third (1.35×10^{-7}) terms, respectively. It is also obvious that this architecture does not meet the loss of control requirements.

The next logical progression is to investigate System 2 as defined by Figure 2 by adding a voter/monitor plane at the sensor/controller interface. This configuration requires no additional hardware since the necessary digital cross strap data link was required for the first system. Although this system has the advantage of being capable of isolating sensor/controller failures from computer failures, it only has a relatively minor effect on overall system reliability. Unless the digital cross strap of sensor information is somehow made independent of the CPU, then a computer failure still has the effect of causing the loss of all sensor/controllers hardwired to it. The additional cross strap does allow the system to inherently service a sensor/controller failure

followed by a processor failure in another branch because all three processors are fully functional after the first failure. However, a processor failure followed by a sensor/controller failure in another branch still causes loss of control under the ground rules. Thus, the addition of the new cross strap reduces the second term in the loss equation by a factor of two (16.90×10^{-7}) and reduces the total probability of loss of control to 20.47×10^{-7} . Unfortunately, this still does not meet the requirements.

System 3 of Figure 2 is produced by providing an analog cross strap of all sensor/controller inputs to each computer. This configuration truly achieves the desired independence of the sensor/controllers from the processors and thus has the effect of reducing the second term of the loss of control equation to zero and the overall probability of system loss of control to 3.57×10^{-7} . Thus, this configuration exceeds the requirements and will probably exceed the goal since this analysis and the inherent assumptions are very conservative. This system architecture is not particularly attractive since each computer must now input three times as much sensor/controller information. In addition, since separation of power must be maintained and a failure in one branch cannot be allowed to cascade into another branch, isolation must be provided on all cross strapped analog signals. Thus, System 3 carries with it a rather significant hardware penalty.

A sensible compromise can be struck between Systems 2 and 3 once it is realized that the major reliability improvement can be retained by only cross strapping the lower reliability sensors in analog form and digitally cross strapping the rest. The assumption that the three sets of rate gyros (pitch, roll, and yaw) and the two sets of accelerometers (normal and lateral) are cross strapped results in the second term of the probability of loss of control equation and the system total to be $.91 \times 10^{-7}$ and 4.48×10^{-7} , respectively. This compromise system still meets the system reliability requirements and incurs only part of the hardware penalties associated with System 3.

At this stage of the development of the system architecture, the probability of system loss of control is being driven by dual like sensor/controller failures and the dual computer failures. Because of this fact, the topics of sensor failures and internal computer architecture will now be addressed.

4. CONTROL LAW RECONFIGURATION

The loss of a second sensor of the same type in a triply redundant system produces a failure state from which recovery is not certain. The detection of such a failure is relatively simple when cross-channel monitoring techniques are used, but the isolation to a particular sensor is difficult. Reasonableness testing and the addition of inflight self-test capability have both been suggested as possible solutions, but these techniques lack the high confidence levels necessary to assure high overall system reliability. The addition of self-test hardware to a sensor will result in the disadvantages of increasing sensor costs, decreasing sensor reliability and adding failure states.

Control law reconfiguration has been developed to take full advantage of a digital FLCC's flexibility and to eliminate the possibility of ever choosing the wrong sensor after a second-like failure.

Flight control system analyses were performed to study the effects of particular sensor losses on the vehicle and to determine if the concept of control law reconfiguration is of value. The system response to a 1-g step input was simulated with individual sensor failures and was compared to the basic response. These individual failures are shown in Figure 3. These responses were retained for a flight condition at Mach 0.9 at an altitude of 30,000 feet.

The system behaved quite differently when either the pitch rate gyro or normal accelerometer failed. The reasons for the predicted divergence are unique to the type of failure. Both of these failures produced an unstable characteristic equation in that the A_n failure caused a lag-term aperiodic divergence of the phugoid mode and that the pitch-rate gyro caused an oscillatory divergence of the short-period mode. The divergent response shown for the A_n failure is exaggerated by the fact that the input to the system is a g-command and feedback is absent. This type of response is acceptable but not very desirable.

The reconfiguration that proved to be most effective for recovery from the A_n failure was to convert the g-command system into a pitch-rate commanded system. As shown in Figure 4, the major change was to sum the stick input upstream of a pitch-rate lag network and to bypass the original washout. The addition of the 10-radian lag stabilized the phugoid mode and increased both the short-period damping ratio and frequency. The step response to an equivalent 1-g command is found in Figure 5.

The loss of the pitch-rate gyro at any flight condition that represents an unstable airframe greatly reduces the chances of simply reconfiguring the system to obtain an acceptable response. Several unsuccessful attempts were made to pursue the possibility of stabilizing the system by changing feedback gains in the acceleration or angle-of-attack paths or both. On the basis of a preliminary analysis, a pitch rate estimator would be desirable. Of several methods developed to predict pitch rate, one was selected because it provided an estimate without differentiating a system state. The method, based on reduced-order estimator theory, requires angle-of-attack and elevator-position feedback as part of a first-order estimator.

The actual estimate is obtained by the application of the following equation to the second-order approximation of the free airframe:

$$\dot{V} = \dot{\theta} - K\alpha \quad (V \text{ is an arbitrary variable})$$

Upon differentiation and substitution of the second-order approximation terms, it can be shown that pitch rate can be eliminated from these equations to yield the following equation, which is linear in angle of attack and elevator position:

$$\ddot{V} - K_1\dot{V} = K_2\alpha + K_0\delta_e$$

As shown in the block diagram in Figure 6, these equations can be implemented to predict pitch rate. An additional advantage of this technique is that the original dynamics of the system are unchanged. In Figure 7, the step response shown is an example of the behavior of the predictor.

The gains K_0 , K_1 , and K_2 are functions of flight condition and are dependent on the value selected for the arbitrary gain K . Values of $K = 10$ and the aerodynamic data, which were available for Mach (0.6, 0.9, and 1.2) and altitude (0, 10-, and 30-thousand feet) variations, were used to determine values of K_0 , K_1 , and K_2 . Several functions of air data were attempted as a means of scheduling these gains; the final selections are presented in Figure 8.

The greatest deviation between a calculated gain and a scheduled gain was approximately 10%. Given this value as a maximum deviation, sensitivity studies were performed on each gain individually and all gains collectively. A summary of the effects of gain sensitivity on short-period frequency and damping ratio are summarized in Table 2 for Mach 0.9 and 30,000 feet. It should be noted that even the maximum deviation from the normal is still within Level 1 specifications.

With the development of the maneuver enhancement mode, it was necessary to define the effects of sensor failures when this mode was engaged. The analysis, performed at Mach 0.9 at 30,000 feet, produced results similar to the basic system response. No significant differences were noticed with the loss of angle-of-attack feedback. The loss of pitch-rate-gyro feedback created a periodic instability, and a loss of normal-accelerometer feedback produced a system that was stable but unsuitable for a system designed to respond to g-commands. The system responses to a 1-g step command for the unfaild as well as the failed cases are presented in Figure 9.

In order to simplify the reconfiguration process, it is desirable to deactivate the maneuver enhancement mode if a second like sensor has failed. Since the intent of reconfiguration is to provide a "get-home" capability, the loss of enhancement modes is not considered to be a penalty. Under this ground rule, the reconfiguration schemes previously described are applicable.

To determine the accuracy of the estimation technique, the pitch-rate response to a 1-g command input was generated and compared to pitch-rate responses derived by two estimators, one using the command input into the integrated-servo-actuator as an approximation to the elevator deflection. The responses, shown in Figure 10 were produced on the TACTIFS digital simulation at 0.9 Mach and at an altitude of 30,000 feet. Exact duplication of the response by the estimator was not expected since the estimator was derived by use of linear aerodynamics and simplified equations of motion. The responses show excellent agreement in the steady state, with the estimators somewhat underpredicting the actual pitch rate in the transient. Thus, use of these estimated results in place of a failed pitch-rate gyro would have the effect of lowering the $\dot{\theta} / \delta_e$ loop gain and thereby cause the system's transient response to become less damped. In an operational vehicle, using a digital model of the ISA, a signal nearly identical to elevator-surface position will be available for generating the estimator signal. All indications point to the proposed pitch-rate estimator as a viable method for control law reconfiguration in the event of a dual pitch-rate-gyro failure.

A lateral-directional failure mode and effect analysis was conducted similar to the longitudinal analysis. Emphasis was placed on the natural frequency and damping characteristics of the dutch roll mode. The findings at Mach 0.9 and 30,000 feet are presented in Figure 11. No instabilities because of sensor failures were expected since the free-airframe characteristic was stable and within Level 1 specifications. However, the loss of a yaw-rate feedback did cause the dutch-roll damping ratio to decrease slightly below Level 1 minimums, which could produce problems for flight conditions where free-airframe damping was low.

A reconfiguration technique for improving this situation has been developed. By removing the A_y feedback and the washout in the $p\alpha$ feedback, as shown in Figure 12, a significant improvement in dutch-roll damping was achieved. The dutch-roll mode characteristics for the normal, sensors failed, and reconfigured cases are summarized in Figure 13.

The reconfiguration technique as applied to dual sensor failures is a viable and attractive alternative to sensor self test. Sensor self test, if successful, does preserve undegraded system performance after a second like failure. However, self test cannot be made 100% successful and does increase sensor complexity and reduce sensor reliability.

With respect to controllers, a somewhat different strategy must be employed since a reconfiguration strategy that simply removes the input in the event of a second like transducer failure does not always result in a safe configuration. A reconfiguration for the pitch stick was considered that allowed control to pass from the sidestick to the twist-grip throttle but was ultimately rejected because this might be confusing to a pilot and the aircraft could be lost before he recognized the condition. Thus, it was concluded that the most prudent approach to isolation of second like controller failures was to employ in-line testing. The most likely controller failure is loss of input, which can occur in a number of ways. This condition is normally difficult to detect quickly, since the most likely output of the transducer is a null, indicating no pilot input. By introducing a known null bias into the transducer output, this situation can be rectified. The bias is set to be larger than the monitor trip level to insure that a null failure usually produces an input transient condition that trips the monitor. The problem of identifying the particular channel failed is greatly reduced since a null input can only be produced by a particular controller position. Thus, the only time that such a failure cannot be quickly identified is during a specific maneuver. This approach avoids the difficulty associated with many detection techniques wherein the system is at a quiescent state as the pilot attempts a reset and, thus, the failure will appear to be a transient failure when in fact it is a hard failure.

Table 3 presents a summary of the failure recovery techniques discussed for a dual sensor/controller failure and also includes these reconfiguration techniques presently employed on the F-16 for angle-of-attack and pilot data failures. As a result of the above discussion, the previously developed probability of loss-of-control equation can now be modified further. Control law reconfiguration eliminates all sensor failures from the second and third terms of the equation. Assuming a confidence level of controller self test (ST_C) of 50% produces the following equation and associated contributions.

$$\begin{aligned}
 P_{loc} &= 3P_C^2 (1-ST) + \sum_{i=1}^3 3P_C P_{Si} (1-ST_C) + \sum_{i=1}^3 3P_{Si}^2 (1-ST_C) \\
 &= 2.22 \times 10^{-7} + .457 \times 10^{-7} + .003 \times 10^{-7} \\
 &= 2.68 \times 10^{-7}
 \end{aligned}$$

Thus, as a result of a properly designed system architecture, the use of control law reconfiguration for dual sensor failures and an assumed modest level of in-line self-test capability for dual controller failures, the loss-of-control probability has been reduced below the design goal. Further improvement is dependent on the inherent reliability of the computer and its ability to perform an adequate self-test function.

5. PROCESSOR DESIGN

Further improvement in system reliability is dependent on the inherent reliability of the computer and its ability to perform an adequate self-test function. Using the reliability model previously defined (including the selected system architecture, control law reconfiguration and a 50% confidence level of in-line self test of controllers) the effects of computer reliabilities and computer self test coverage can be observed. Figure 14 presents the probability of loss-of-control as a function of computer mean time between failure (MTBF) and self-test confidence level. From this figure, it is evident that although significant improvement can be made, i.e., achieved in reducing the probability of loss of control, that an order of magnitude improvement is the practical limit for present technology.

There are several desirable features which can be designed into the computer which reduce the likelihood of a single internal computer failure resulting in a total branch failure. From Table 1 it is observed that the processor and the memory have the highest failure rates. Thus, one way to prevent the loss of dedicated sensor information to the other branches of the computer complex is to permit the I/O controller and serial digital data links to operate independently of the processor. This requires that the I/O controller operate from a controller sequence such that the processor normally controls which I/O functions are being performed. However, when the processor fails (possibly indicated by a watchdog timer associated with the I/O controller) the controller continuously inputs sensor/controller data and transmits it via the serial link via DMA to the other processors. Thus, triplex input data is preserved when less than three processors are operating. In order to reduce the effects of memory failures, it is desirable to segregate computational functions (pitch, roll and yaw calculations) with no common code or subroutines used in more than one partition. Thus, the failure of a pitch computation in one computer and a roll computation in another computer would be treated as two dissimilar first failure rather than as a second like (memory) failure. The reliability of the computer can also be enhanced by the addition of various system monitors.

5.1 System Monitors

Monitors of various forms (software and hardware) can be employed to enhance failure protection and to aid in the isolation and identification process. Care must be exercised to prevent over-monitoring, which might result in a failed but usable subsystem being rejected in flight, or under-monitoring, which might produce a latent failure condition. The subsystems and devices which should be monitored include the processor, input subsystem, output subsystem, and inter-FLCC interface.

5.2 Processor Monitoring

Because of its complexity, the FLCC processor subsystem (Central Processing Unit, memory, and controllers) is the most unreliable element in the DFBW system from a series MTBF standpoint. Therefore, the failure monitors associated with the processor must interact with the redundancy management process in a fault-tolerant manner. The voter/monitor algorithms are designed to detect only those failures which would significantly affect the processor output. These algorithms are all that are required to protect the system from a first failure and to detect a second like failure. It is only when one must isolate a second like failure that additional monitors are needed. These additional monitors are designed to detect all failures to a very high confidence level, and thus, to detect failures which do not significantly affect processor output as well as those which do. When all three FLCC processor systems are operational, this additional failure information is useful for subsequent maintenance, but should not be used to terminate a defective but functioning processor. The above rationale was used to establish design criteria with respect to processor monitors, as follows:

1. When three processors are operational (outputs are within tolerance), processor monitors which duplicate the failure coverage provided by the voter/monitors will be used for information purposes only.
2. When two processors are operational, processor monitors will not be permitted to declare a failure until the two corresponding outputs from the processors are not within tolerance.
3. When only one processor is operational (no backup being available), processor monitors will not be permitted to disengage the system.
4. All failures detected by the processor monitors will be recorded in nonvolatile memory.

As with all criteria, these have exceptions, which will be discussed in the following paragraphs. Five types of inflight monitoring are used to detect and isolate processor failures; a watchdog timer, CPU self test routine, memory parity checks, overflow limiter, and power supply monitor.

Watchdog Timer. The watchdog timer is a monostable switch that must be periodically pulsed (strobed) in order to prevent the presence of a discrete input into the "I AM GOOD" logic which feeds the other processors. On "power up", the timer does not become active until the first strobe so as to permit all processors to get on line. Once activated, the timer must be strobed to prevent a failure indication. The timer is used primarily to detect processor failures which either halt the processor or put it into a loop. These failure types are detected by other system monitors when three systems are operating. However, when only two processors are operating, the timer is needed to ensure that the failed processor is disengaged. This is necessary because the redundancy management scheme does not permit one processor to shut another one down. A processor may only be shut down by itself or by both of the others declaring it failed. Since this monitor is external to the processor, there is no easy way to make it conform to the design criteria. Therefore, this monitor will be allowed to operate regardless of the state of the system.

The software which strobes the timer should be so located that, if the monitor trips, the disconnect will occur at about the same time as the output vote occurs. This is done to minimize any transients associated with holding current outputs until a failure has been isolated. The software should also be simple to execute and be protected by some form of a jump command to prevent the processor from exciting the code inadvertently. The tendency to require some fancy form of data manipulation to be performed prior to the output strobe should be avoided since this tends to be a CPU test and thus duplicates the function of another monitor.

CPU Self Test. The CPU self test is designed to monitor the proper operation of the arithmetic unit, instruction set (microcode), memory controller, the portion of main memory containing the self-test program, and the registers used for computation, counting, and storage. A complete testing of these devices is impractical in flight because of the almost limitless combinations of inputs that would be required. Fortunately, several factors contribute to permit an excellent self-test to be performed in flight. The first major contributor is having time available to perform the test. This is accomplished by performing the test during data acquisition, which is under DMA control. It is estimated that data acquisition will require about three milliseconds, which permits a fairly comprehensive self test to be accomplished. The second factor, which also adds time, is that the redundancy management task is reduced when only two processors are operational, thus producing more idle time at the end of the iteration. This time can be utilized for additional testing in a background mode. The third factor is that the most likely failures are stuck bits, which are easily detected by forcing all lines and registers in both states. One must also remember that the redundancy management scheme does not use these tests in the decision-making process until after a second like failure has occurred. The voting algorithms are designed to continue outputting the previous calculation until such time as a failure is detected by self-test or a predetermined time has expired, in which case an arbitrary choice is made. Under these circumstances, several iteration times can pass before even an unstable aircraft is beyond recovery. Thus, more than sufficient time exists to perform a comprehensive test.

Memory Parity. The memory parity test will employ a single-parity bit on all memory to detect all odd numbers of bit failures in a given memory word. Assuming an equal likelihood of any bit failing, the probabilities of multiple bit failures in a single word is infinitesimal during a given flight. Multiple bit errors are easily detected by memory sum checks between flights, or by the voter/monitors if the errors occur while three processors are operational. Memory failures constitute approximately one-half of all processor failures, and the location of the failure is of importance in minimizing its effects.

The type of memory used in the processor also influences the response one must take to a failure indication. For example, if core memory is used, one cannot ignore a parity error if the memory parity bit is regenerated on each write cycle. If a bit were changed, the parity would be corrected so that on the next access of that word there would be no parity error. This is not a problem with ROM and RAM memories, and the parity error would be redetected on each subsequent pass. This property was one of the deciding factors in the selection of memory types for this application. Once it is guaranteed that the error will be redetected, one can again rely on the cross-channel monitors to distinguish between failures of significance and insignificance.

The strategy selected is to ignore parity errors as long as three channels are good, or when two channels are good and their outputs agree, and to use parity to aid in the selection of a good channel when the two outputs disagree. In all cases, parity errors cause an interrupt to be generated and the location of the error to be retained. This is a form of fault tolerance for noncritical failures.

Overflow Limiter. The DFEW processor will have overflow limiting capability either by supplying firmware (add, subtract, divide, and left shift) or by providing an interrupt that will be serviced to produce a limited output. This feature can be used to isolate software faults for later correction but must not be used in the redundancy management process. The most likely cause of overflow is a common software fault produced by inadequate scaling. Such a fault could cause all three processors to shut down. The approach is to detect the overflow, limit the result to the maximum value of the proper sign, and continue the program. During software development and system tests, the program will be executed with the limiting feature disabled so that all overflow conditions will be detected and intelligent decisions made in each case as to whether the software should be rescaled or whether the limiter feature should be utilized in this particular case.

5.3 Power Supply

Since the inputs and outputs to the FLCC power supplies are predetermined, failure monitoring is relatively straight-forward. Monitors will be employed on each output voltage level to detect voltages below expected values. The outputs will be logically OR'ed so that any supply voltage failure is flagged via a discrete to each CPU. Since any power supply loss must be considered fatal, the receipt of a failure indication from another processor will be all that is required to declare the processor bad. In order to prevent transient failures from becoming permanent, the software will automatically declare a processor OK if power is subsequently restored. This is identical to the problem one has on "power up" since a processor will have to permit another processor to get on line if nothing besides a power monitor failure had been detected.

5.4 Input Subsystem

The analog input subsystem is provided with several hardware augmentations to provide simplified functional testability. These consist of fresh data flags, A/D scaling checks, DMA addressing checks, and through-put monitoring. When used in conjunction with software routines, these test functions increase confidence in data validity and subsystem operation.

Fresh data flags are provided by inserting a defined bit pattern into the unused 4 bits of the 16-bit data word containing 12 bits of digitized analog data. The data destination in scratch-pad memory will be initialized prior to data acquisition to provide a contrasting data pattern for these 4 bits. A monitor of these word segments after the data transfer will provide evidence of a successful transfer by DMA of the digitized data.

Analog-to-digital conversion scaling checks will be provided by monitoring of the system power buses as analog input channels. The voltages will be scaled to provide inputs to monitor A/D digitizing checks within the reasonable accuracy ranges of power supply and converter tolerances.

The application of the power supply monitors to the multiplex inputs of the A/D will place these predictable values in predetermined memory locations. Judicious selection of these memory destinations will provide confidence checks of the address control of the DMA transfers.

The fresh data flags will also support a software monitor of analog-to-digital conversion throughput. Since data acquisition will occur by DMA and will be transparent to the processor, other tasks can be executed during this time. By monitoring the fresh data bits, the completion of the data acquisition cycle can be tested.

5.5 Output Subsystem

A large amount of hardware with limited testability separates the processor from the control surfaces. Detection and recovery from failure requires the addition of output monitoring capability to the FLCC system. To accommodate this task, analog inputs will be provided to supply the processor with a monitor of the current drive of each servo-valve being driven by that FLCC. This single monitor per output provides a reasonable detection of any electrical failure in the complete ISA output system.

In addition to electrical failures, a monitor of the ISA itself is provided through feedback of the integral position transducers. This analog input, when supported by a software model of the ISA, will enable confirmation of proper function by the ISA.

5.6 Intercomputer Interface

The achievement of system reliability depends heavily on the redundancy provided by the inter-FLCC communication buses. Because of the reliance on this data exchange, a high confidence level must be associated with the integrity of the data. Hardware monitors of this serial data link include word parity, word sync, and message-length monitors.

Data transmission will utilize a Manchester bi-phase-level code, including word synchronization bits and odd-word parity. The characteristics of this self-clocking code provide a high level of bit error detection and noise insensitivity. Data-receiving circuits will detect data words of incorrect parity or improper length and provide a flag to the processor to indicate detection of these errors. Data transmission will consist of known message lengths and will be monitored by the receiving hardware to identify short or long messages. Again, the status of this monitor will be made available to the processor as an error flag.

A monitor completely written in software, included here for completeness, is the transmission of a known-value data word for message confirmation. Since depositing of this data into memory is through a DMA channel and transparent to the software, a double check of message completeness and addressing correctness is desired. Since most of this coupled data will include full utilization of the 16 bits of the data word, a fresh data flag for every word as used for analog input data is not appropriate. The known-value data words will, however, provide system checks similar to the analog input of scaled power supply voltages, with the advantage of zero-tolerance requirements on data content.

6. ACTUATOR INTERFACE

The analysis thus far has neglected the redundancy management considerations associated with the actuation system and its interfaces with the flight control computer complex. The primary reason for this is the fact that the most probable failure modes associated with the actuation system are the loss of hydraulic pressures (the computers having no means of recovery from these failures). In addition, the inclusion of the actuation system would unnecessarily complicate the analysis conducted previously.

The actuator chosen for the DFCS analysis was the F-16 integrated servoactuator (ISA). The F-16 ISA is an integrated version of the servos and actuators employed on the F-111 and has a proven reliability record. Originally designed to a fail-operate/fail-safe criterion on the F-111 using electronic and hydraulic selection with three inputs, the concept was upgraded to a fail-operate/fail-operate/fail-safe criterion by use of a fourth channel of electronics as an active back-up on the statically unstable F-16.

The ISA has two coils on each of three valves which can be driven independently to provide a variety of interface combinations. The ISA can also be forced to operate from the primary valves (1 and 2) or the secondary valve (3) using external commands. The net result of these features is an ISA which can be configured to meet the DFCS requirements in several manners. One of the design tasks was to determine which if any of these several configurations offered advantages from a reliability, complexity, or cost standpoint.

For the purpose of analytical simplicity it is assumed that the ISA's are driven by three identical sets of electronics (computers, sensors and associated interfaces) and that this system, exclusive of the ISA's is capable of meeting the loss of control goal (3 failures in 10^7 flight hours) and that the system as a whole is capable of second failure recovery to a confidence level of 95%. Thus the probability of electronic system failure can be expressed as:

$$P_e^3 + 3P_e^2 (1 - P_e)(1 - .95) = 3 \times 10^{-7}$$

Solving the equation for P_e produces the result that $P_e = .0014086$. Assumptions for the reliability of the hydraulic systems are based on a 3500 hour MTBF for the primary system (A) and a 2500 hour MTBF for the secondary system (B) and a 1.4 hour flight time. Thus the probability model for analysis consists of three electrical systems and two hydraulic systems. Table 4 shows the thirty-two possible failure configurations (a "0" indicates system failed) and the associated probabilities of the total system being functional

irrespective of self test confidence level. It is assumed that there is zero probability of safe recovery after either three electronic failures or two hydraulic failures. Thus the total system has a maximum probability of success of .9999997776 and a minimum probability of failure (loss of control) of $.2224 \times 10^{-6}$.

It is assumed that electrical failure detection is 100% first failure through the use of voting and cross-channel monitors within each electrical system and that each system has hydraulic system selection capability by activation of ISA reset solenoids. Electrical recovery on second failure assumes self-test detection and the ability to activate the configuration's back-up. Hydraulic recovery probabilities are based on success of the internal ISA voters and assume both hydraulic systems are functional. The composite recovery probability is the combination of recovery probabilities based on the equation

$$P = (\text{Prob of elect recovery}) + (\text{Prob of hyd recovery}) \times (\text{Prob of electrical recovery unsuccessful})$$

Configuration 1, presented in Figure 15, displays two servo amplifiers in each FLCC. One amplifier provides current to the primary coil of its associated servo valve and the second is used to backup the amplifier of another branch. This configuration has the disadvantage of requiring six amplifiers but is very tolerant of their failures. The configuration can survive six of the nine possible combinations of two electronics and one hydraulic failure to a .975 confidence level. This level is achieved by combining the 95% confidence level self test with a coin flip.

Configuration 2, as shown in Figure 16, accomplishes the ISA interface using only three amplifiers. Branch A drives three coils with branches B and C driving two and one coil respectively. As seen in Table 4, this configuration is capable of recovering from seven of the nine critical failures to a .975 confidence level. The disadvantage of this configuration is that the three servo amplifiers are driving different loads and thus the three processors would have to be different and thus not interchangeable.

Configuration 3 (Figure 17) employs an exterior voter/selector (smart switch) and a fourth amplifier on an active/standby basis to accomplish the interface. Under the assumption that the D branch is selected only by the first failure, the interface is capable of recovery from eight of the nine critical failures with a confidence level of .975 and .4875, depending on which hydraulic system is functioning. Adding additional intelligence to the voter/selector to permit amplifier D to be reassigned after a second failure would permit the confidence level to be raised to .975 for all cases. The configuration has the apparent disadvantage of requiring the addition of a fourth LRU to the system and another power supply. Although this is true, it is also frequently necessary in redundant systems to provide a "home" for elements of the system which either don't require redundancy or cannot, because of practical design considerations, be made redundant.

Configuration 4 is a variation of Configuration 2 which achieves the desired goal of balancing the load on the three servo amplifiers. This configuration, however, did not permit a successful recovery when only computer A was good and hydraulic system A had failed (see Figure 18) and thus resulted in an overall reliability penalty.

The system shown in Figure 19 is a variation of Configuration 3 which avoids the necessity of adding a fourth LRU to the system. This is accomplished by the addition of a hardware (analog) voter in each branch and cross strapping the system outputs in analog form. The voter must be supplemented with additional intelligence from the digital processors in order to properly function after a second electronics failure. The design has a disadvantage in that the voters must be powered independently from the branch electronics in order to preclude a dual power supply failure from causing loss of control.

The relative invariance of the loss of control probabilities (Table 4) for the five configurations prompted the analysis of a sixth configuration. This configuration shown in Figure 20 was postulated to answer the question as to how unreliable the total system could become if no special care was taken in the actuator interface. The net result was a seventeen fold increase in the loss of control rate as indicated in Table 4.

The results of the actuator interface analysis indicate that the overall reliability of the DFCS design was relatively invariant as long as proper attention was paid to providing a means of recovery from the loss of two electronic branches and a hydraulic system failure. Thus the question as to what actuator interface is the most practical should be decided by an analysis which deals with the questions of hardware and software complexity rather than with basic architecture.

7. CONCLUSIONS

Based on the analysis summarized herein, the following conclusions can be drawn:

1. A triplex digital flight control system (DFCS) can be designed to meet the loss of control reliability requirements associated with modern fighter aircraft.
2. Cross strapping lower reliability system inputs in analog form provides a significant reliability improvement.
3. Control law reconfiguration solves the dual like sensor failure problem and enhances system safety.
4. Digital computer reliability and self test confidence levels are areas where significant improvement in reliability can still be made.
5. Hydraulic system failure rates dominate the reliability equation associated with actuation and thus actuator interface design should be based on hardware/software complexity trades rather than reliability.

REFERENCES

1. AFFDL-TR-79-3016
AFTI-16 Predesign and Preliminary
Development of DFCS and HAC
Volume II
Task II - DFCS Preliminary Development

LIST OF SYMBOLS

A/D	Analog to digital
A_n	Normal acceleration
ARI	Aileron to Rudder Interconnect
A_y	Lateral acceleration
BIT	Built-In-Test
CCF	Control-Configured Fighter
CCV	Control-Configured Vehicle
CPU	Central Processing Unit
CS	Cross Strap
D/A	Digital to Analog
DFEW	Digital Fly-By-Wire
DFCS	Digital Flight Control System
DMA	Direct Memory Access
DOF	Degrees of Freedom
DR	Dutch Roll Damping Ratio
EMC	Electromagnetic Compatibility
FBW	Fly-By-Wire
FLCC	Flight Control Computer
i	Summation index
IFIM	Inflight Integrity Monitoring
I/O	Input/Output
ISA	Integrated Servo Actuator
LRU	Line Replacement Unit
LVDf	Linear Variable Differential Transducer
M	Mach
MTBF	Mean Time Between Failure
MTBFF	Mean Time Between First Failure
p	Roll Rate
P_c	Probability of Computer Failure
P_e	Probability of Electronic System Failure
P_{loc}	Probability of Loss of Control
P_s	Probability of Sensor Failure
P_s/P_o	Ratio of system pressures used for gain scheduling
q_c	Impact pressure
RAM	Random - Access Memory
ROM	Read - Only Memory
s	Laplace operator
S_p	Short Period
ST	Self Test Confidence Level
ST_c	Self Test Confidence Level for Controllers
TACTIFS	Tactical Integrated Flight System
Σ	Summation
α	Angle-of-attack
$\dot{\theta}$	Pitch rate
δ_e	Incremental elevator surface deflection
$\dot{\psi}$	Yaw rate
ω	Undamped natural frequency

TABLE 1 COMPONENT RELIABILITIES

<u>COMPUTER</u>		1,150
Computational Unit		1,430
I/O Control	9,500	
Processor	3,750	
Memory Control	9,500	
Memory (20K)	4,500	
Power Supply	25,000	
Analog Input	15,000	
Analog Output	30,000	
Cross Strap	35,000	
<u>SENSORS/CONTROLLERS</u>		
Gyro (Pitch, Roll Yaw)	12,000	
Accelerometer (Normal & Lateral)	32,000	
Stick Force (Pitch & Roll)	150,000	
Pedal Force	220,000	

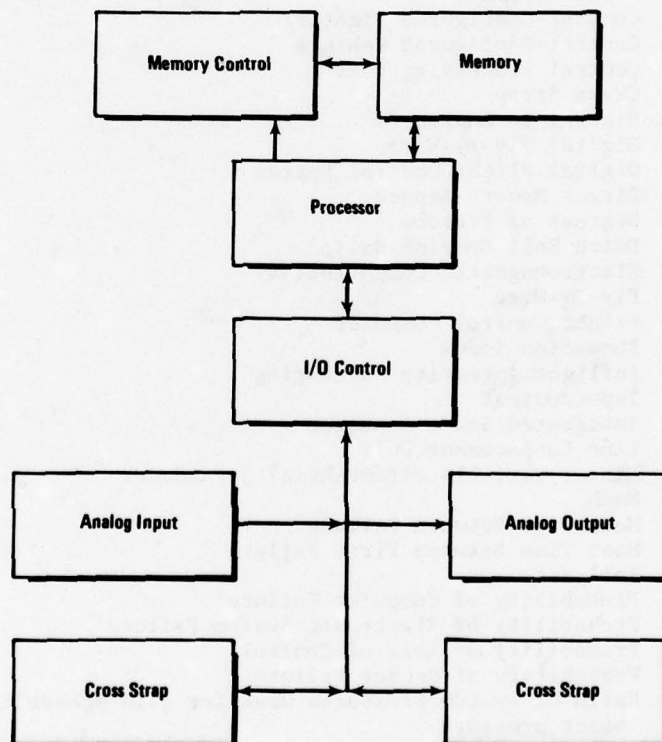
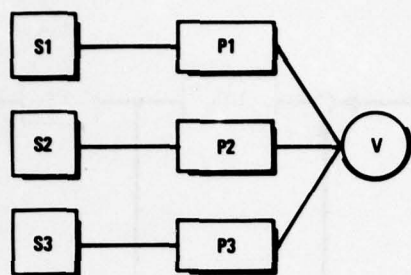
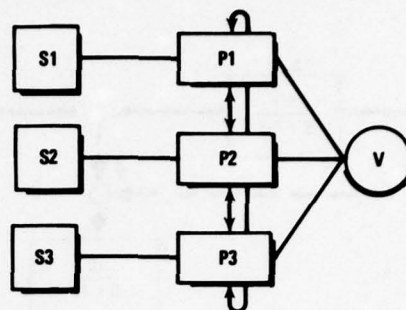


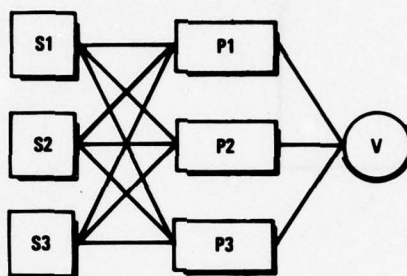
Figure 1. FLCC for Reliability Analysis



System 1 In-line



System 2 Cross-Strap



System 3 Multi-input

Figure 2. Three Candidate FCS Configurations

M = 0.9/30K

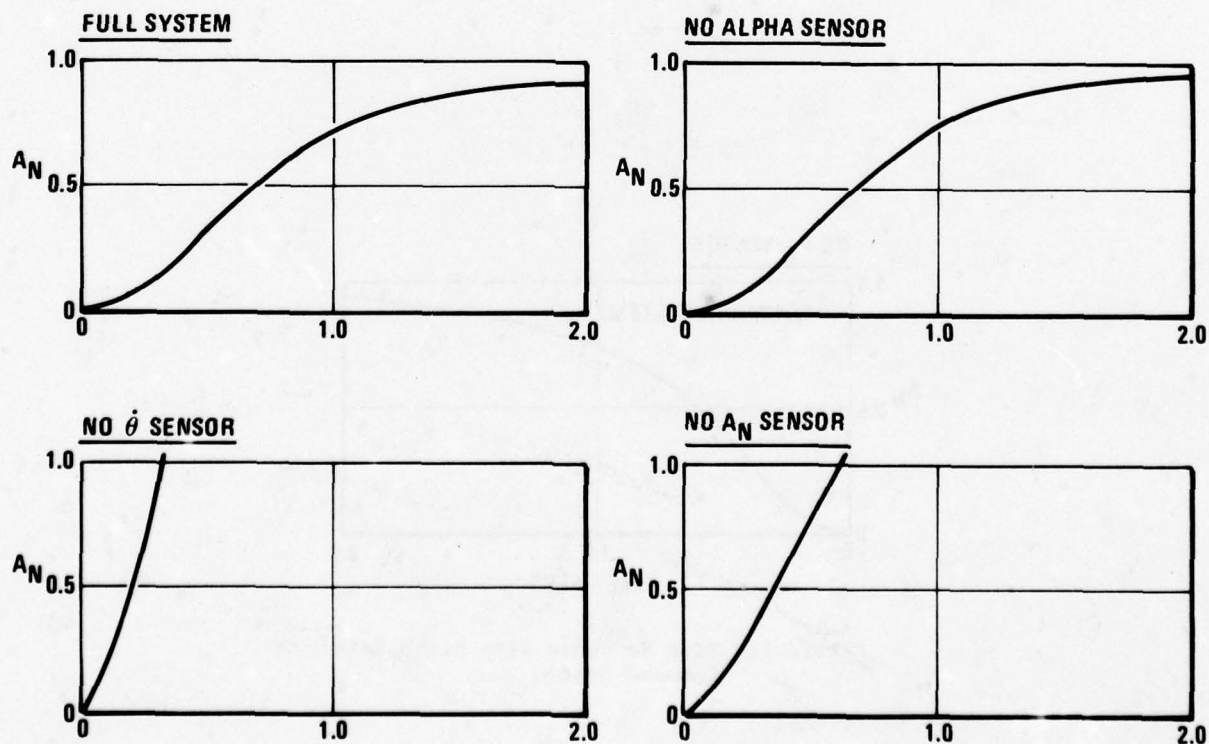
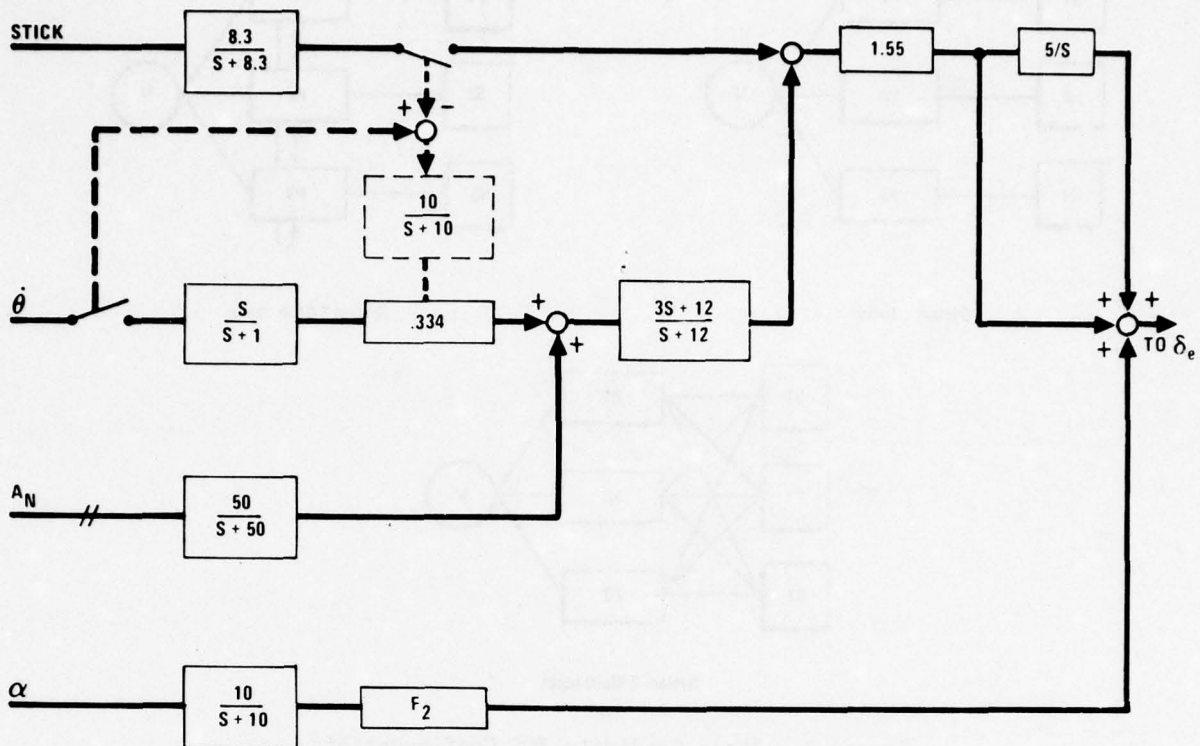


Figure 3. System Response to 1-g Step



A_N SENSOR FAILURE - \diamond SWITCH TO BROKEN PATH

Figure 4. A_N Sensor Failure - $\dot{\theta}$ Command System

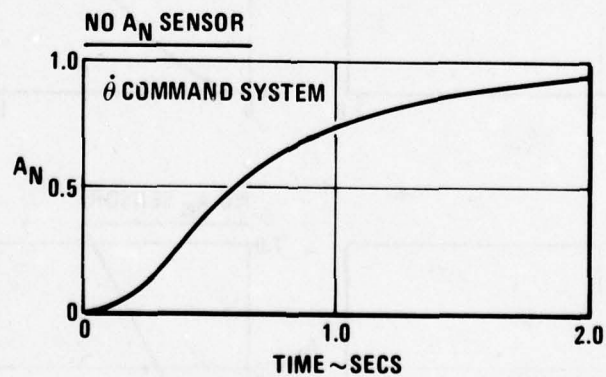
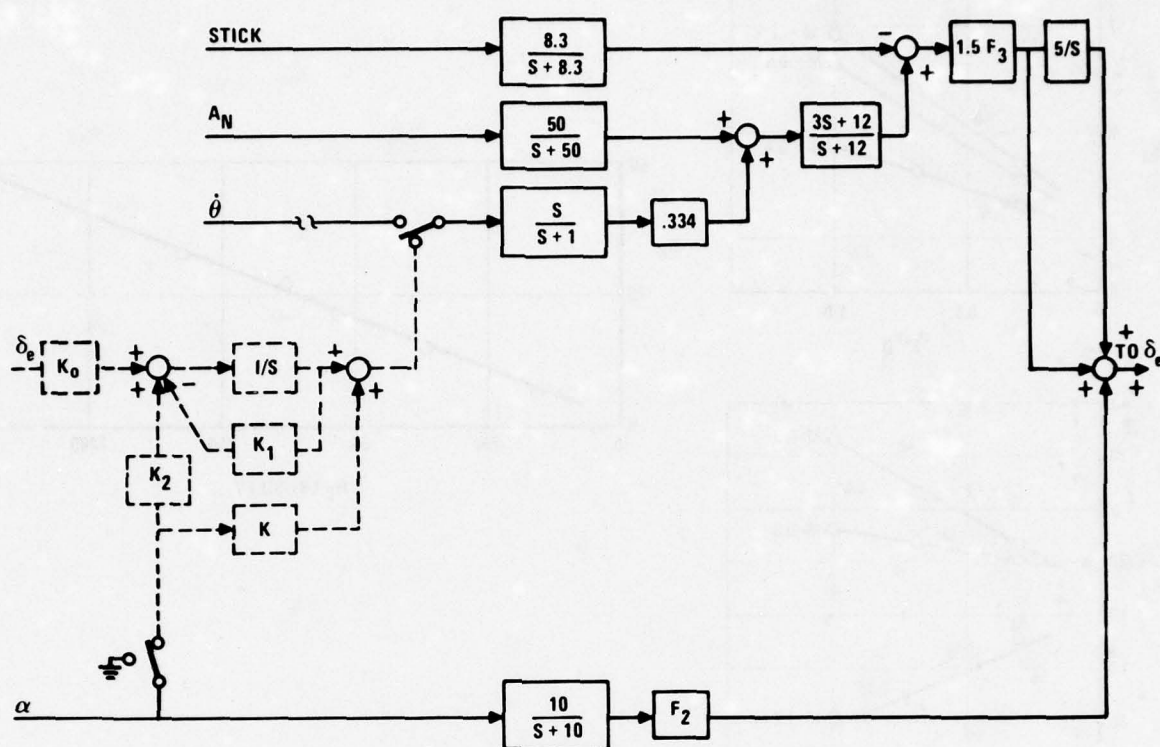


Figure 5. Step Response with Pitch Rate Command System



$\dot{\theta}$ GYRO FAILURE \rightarrow SWITCH TO BROKEN PATH

Figure 6. $\dot{\theta}$ Gyro Failure - $\dot{\theta}$ Estimator

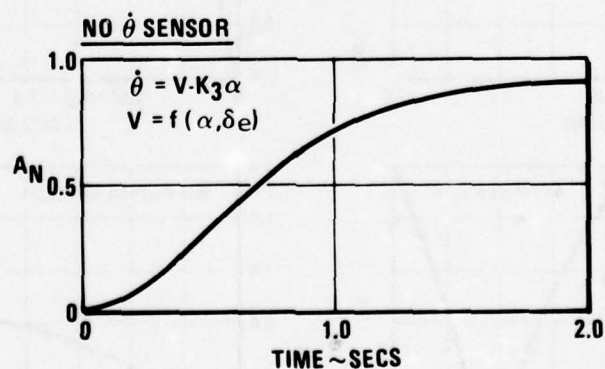


Figure 7. Step Response with Pitch Rate Estimator

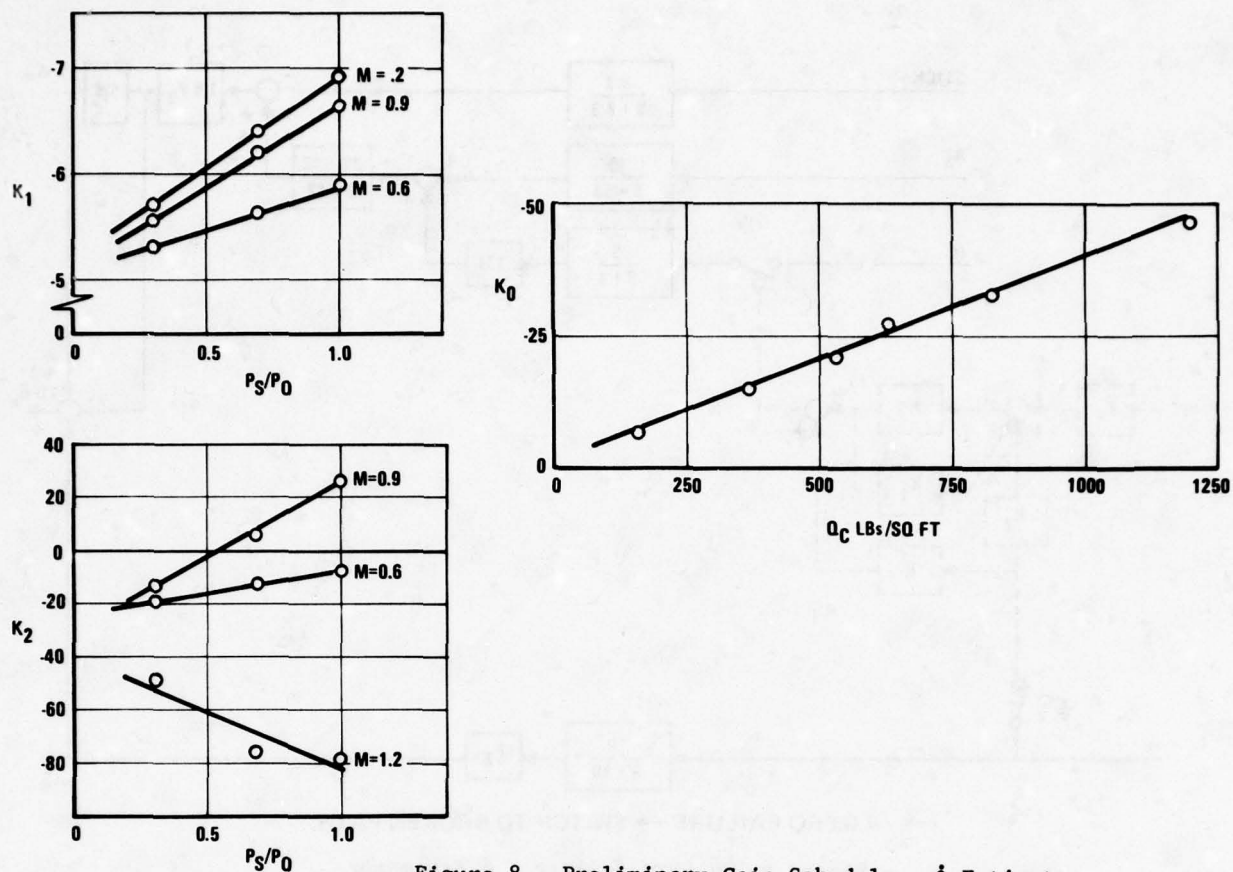
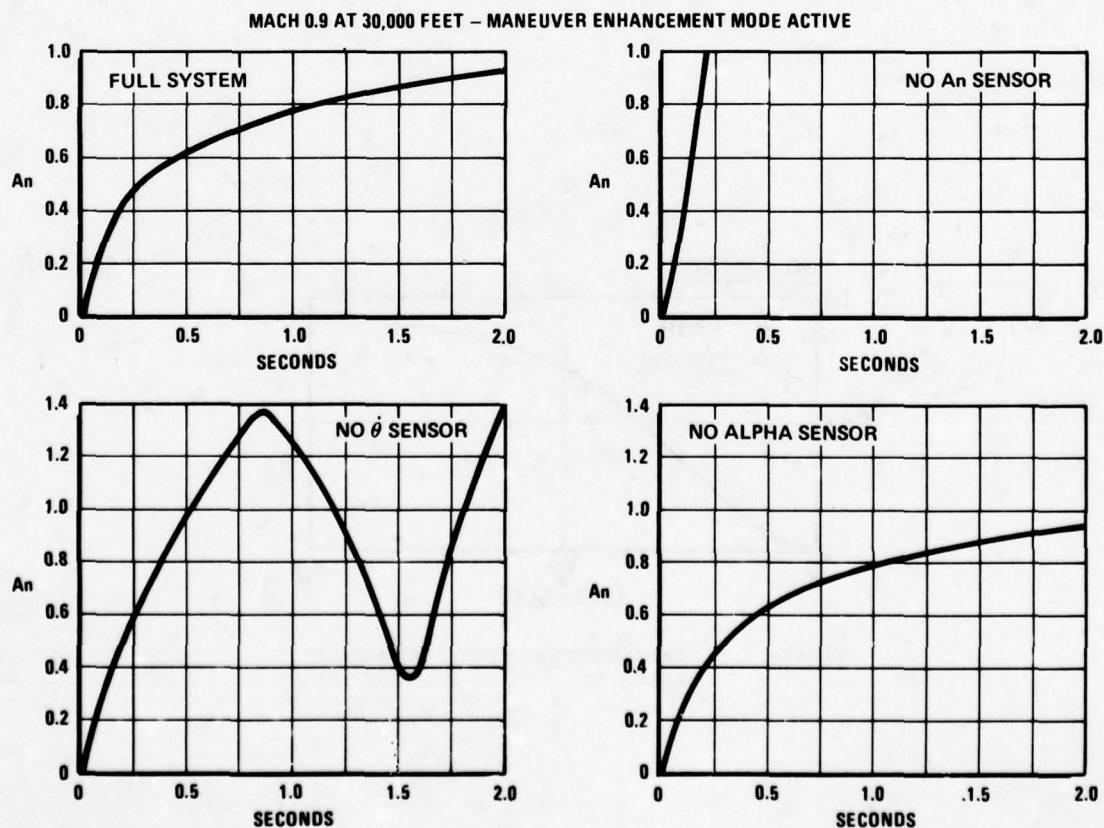
Figure 8. Preliminary Gain Schedule - $\hat{\theta}$ Estimator

Figure 9. System Response to a 1-g Step

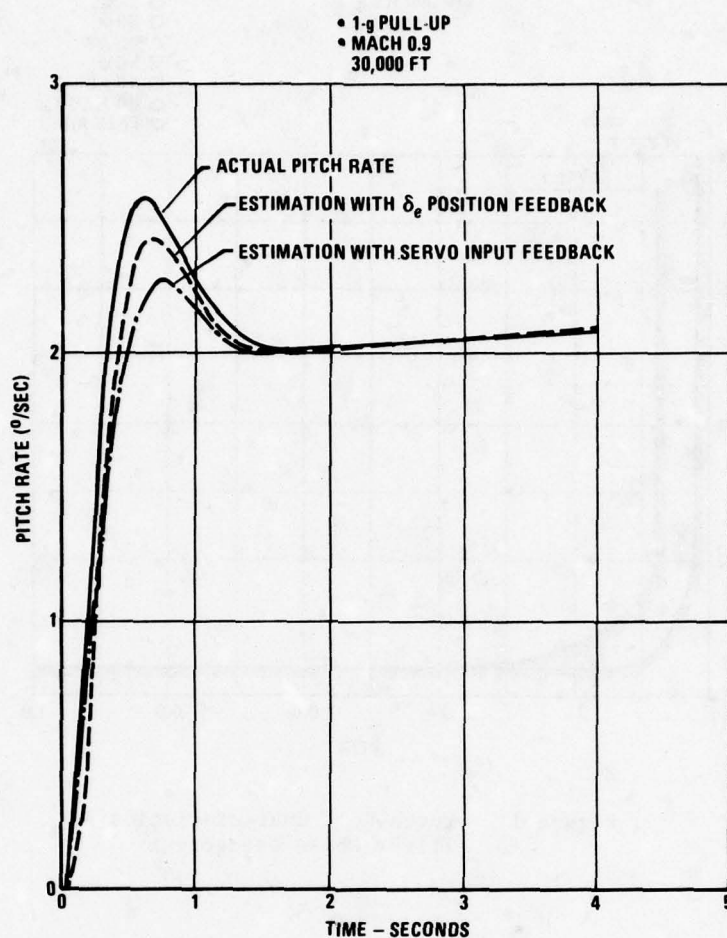


Figure 10. $\dot{\theta}$ Estimator Response Comparison

TABLE 2
LONGITUDINAL SHORT-PERIOD CHARACTERISTICS
FOR GAIN VARIATION OF -10 PERCENT

	Nom	K_0	K_1	K_2	K	A11
ω_{sp}	4.245	4.400	4.035	4.854	3.669	3.958
ξ_{sp}	0.728	0.729	0.882	0.654	0.812	0.854

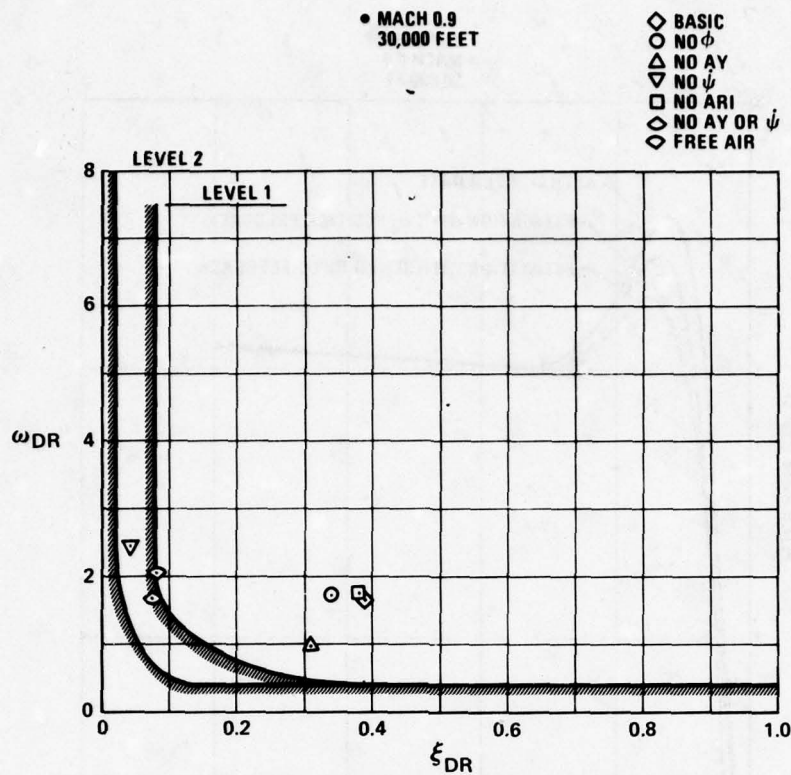


Figure 11. Dutch-Roll Characteristics -
Flight Phase Category B

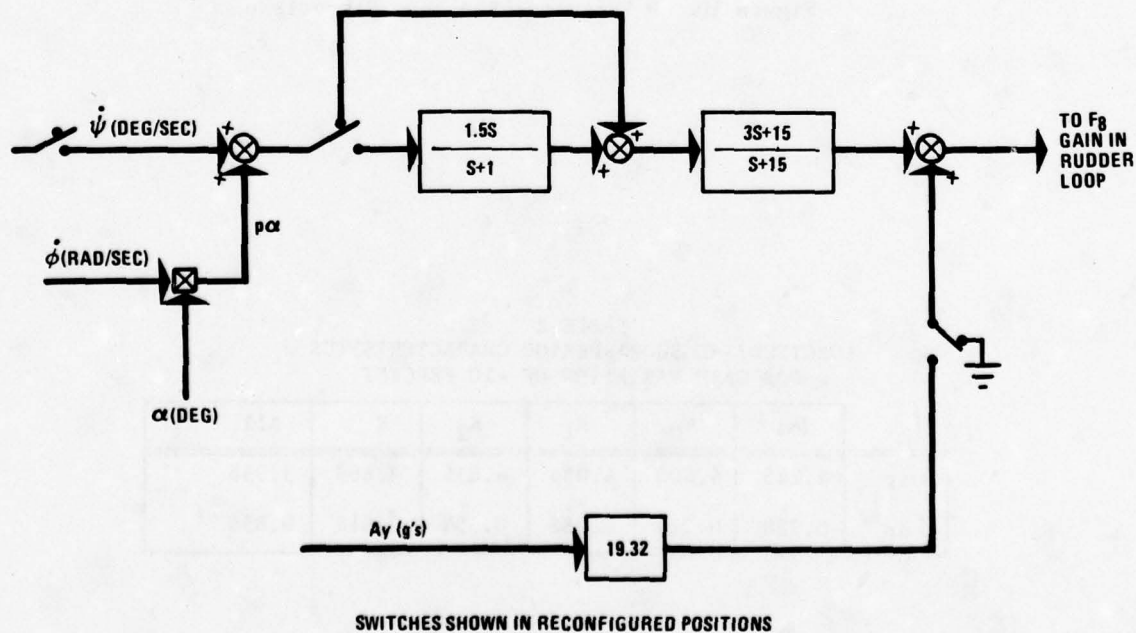


Figure 12. Reconfiguration Technique for
Dual $\dot{\psi}$ Failure

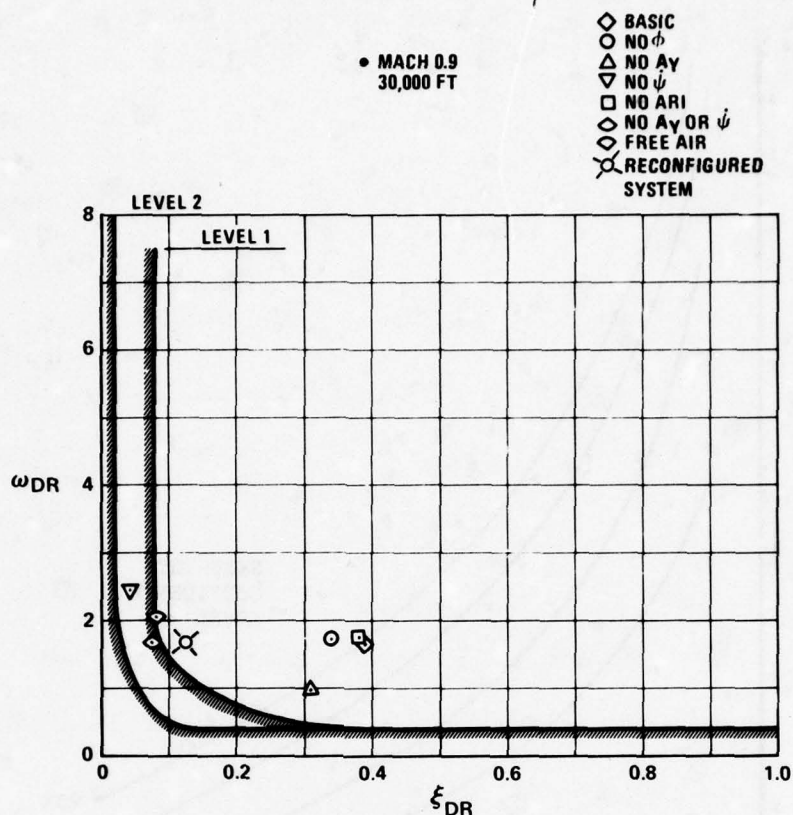


Figure 13. Dutch-Roll Characteristics -
Flight Phase Category B

TABLE 3
SUMMARY OF DUAL SENSOR/CONTROLLER FAILURE
RECOVERY TECHNIQUES

SENSOR	FAILURE RECOVERY PROCEDURE
Pitch Rate Gyro	Estimate based on AOA and horizontal tail position
Normal Accelerometer	Reconfigure to $\dot{\theta}$ command
Angle of Attack (AOA)	Remove feedback
Pitot Data	Use fixed gains
Pitch Stick Transducer	Use in-line test
Pitch or Roll Trim	Use alternate trim
Roll Rate Gyro	Remove feedback, adjust stick gain
Aileron Stick Transducer	Use in-line test
Rudder Trim	Remove input, center trim
Rudder Force Transducer	Use in-line test
Lateral Accelerometer	Remove feedback
Yaw Rate Gyro	Remove A_y and $\dot{\psi}$ feedbacks, use only $p\alpha$ feedback to rudder

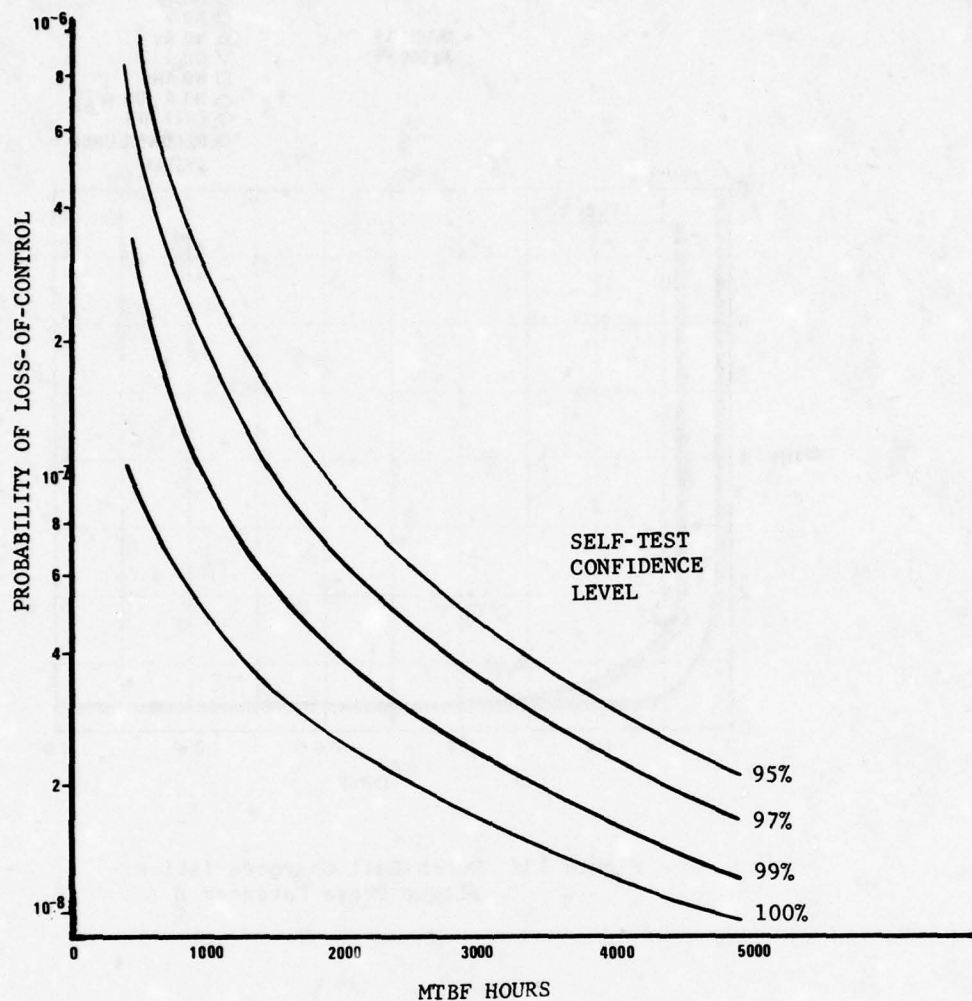


Figure 14 Effects of Computer Reliability and Self-Test Confidence Level on Loss-of-Control

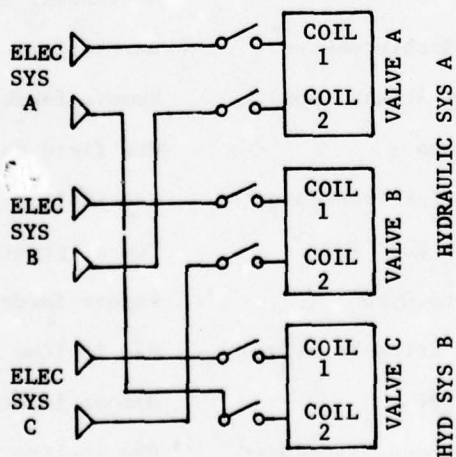
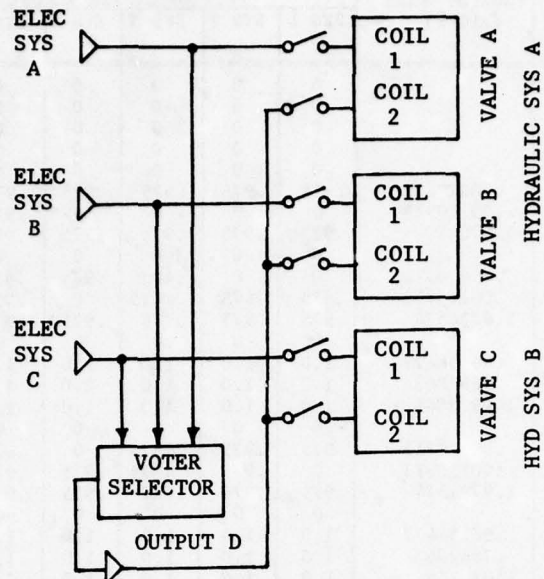


Figure 15. ISA Interface Configuration 1



OUTPUT D OPERATIONAL WHEN ANY TWO SYSTEMS GOOD
AND 95% WHEN ONE SYSTEM GOOD. OUTPUT D SELECTED
BY FIRST FAILURE ONLY.

Figure 17. ISA Interface Configuration 3

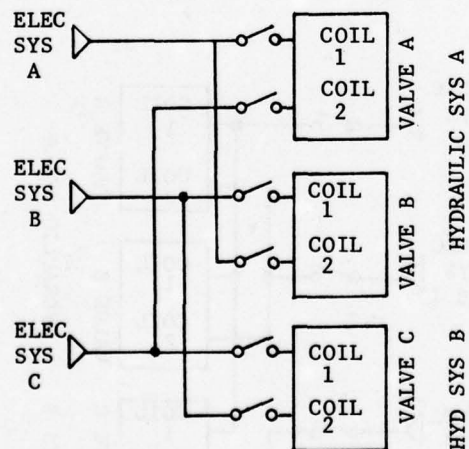
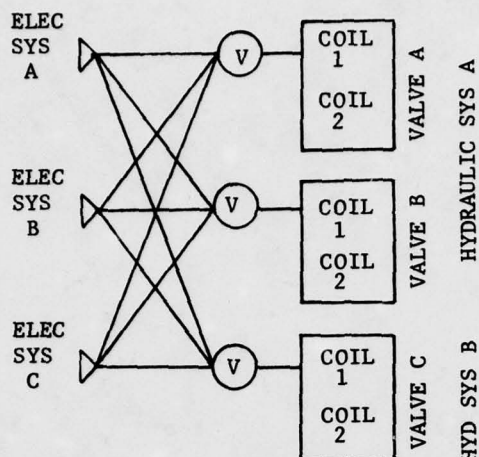


Figure 18. ISA Interface Configuration 4



VOTER SELECTS OUTPUT BY 1 Voting for First Failure
 2 Output Selection Based on
 Self Test After First Failure

Figure 19. ISA Interface Configuration 5

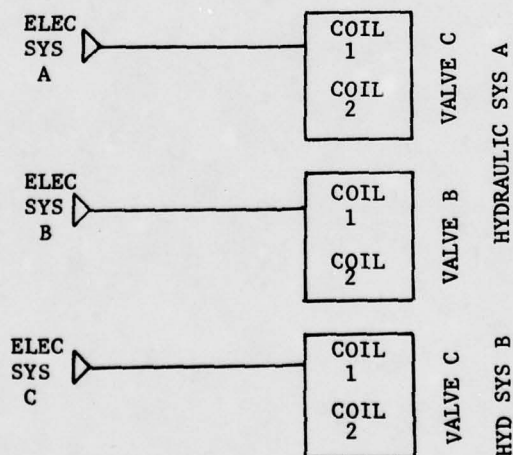


Figure 20. ISA Interface Configuration 6

FAILURE DETECTION, ISOLATION AND INDICATION IN HIGHLY INTEGRATED DIGITAL GUIDANCE AND CONTROL SYSTEM

by
Dr. Wolfgang J. Kubbat
MESSERSCHMITT-BÖLKOW-BLOHM GMBH
AIRCRAFT DIVISION
8000 München 80, Postfach

SUMMARY

The paper covers a broad spectrum of modern failure detection and isolation techniques. It starts with clear indications that the failure problem can be significantly reduced with technology and design.

Several advanced methods such as vector redundancy, dissimilar redundancy, and methods applied to computers are described and some are backed up by practical examples.

The final part is dedicated to data bus orientated guidance and control systems. Based upon a practical realization example are guidelines for the use of MIL STD 1553 B in redundant applications layed down.

1. INTRODUCTION

With the advent of more and more fly-by-wire projects, CCV-aircraft and Active Control approaches one sometimes gets the impression that many airplanes are built just in order to get electronics airborne.

It stresses the important role of new electronic and system technologies, but some clarification is needed.

In the past, various improvements of the aircraft performance have been achieved in aerodynamics, engines, structures and materials.

However, the widening of the aircraft's envelope to higher mach numbers and altitudes has presented the first absolute need for electronics. It was impossible to cover the changes of the aircraft's behaviour which go with mach number and altitude with pure configuration measures. The invention of electronic aids such as Stabilisation Augmentation Systems (SAS) and Autopilots became unavoidable.

In fact, these aids helped to extend the flight regime more and more and suddenly one realized, that these electronics became more important purely from the safety aspect than one originally had intended.

On the other hand, since especially the military environment has become more and more competitive, it is impossible to backtrack on these developments.

It seems that presently the most significant improvements in the performance of new airplanes are based upon the use of electronic aids. Additionally new possibilities of control will be given to the human pilot, greatly enhancing the aircraft's agility. But he certainly is not able to utilize them without electronic aids.

This clarifies that equipping an airplane with electronics is not a purpose in itself but an indispensable prerequisite for progress in performance and agility today.

On the other hand, electronics presents a problem in itself, cost and reliability. And last but not least there is the psychological problem of switching over from what was well known for almost a century to the absolute dependance on electronics.

Facing the fact that there is no "back to nature" there are many ways to solve the reliability problem and clear indications are present that even in the long run the cost will be less than with conventional systems. Finally there remains only the psychological syndrome. Time and experience will cure this.

2. PROBLEM DESCRIPTION

- We assume that the aircraft has to be equipped with electronics.
- The aircraft's safety will depend on the proper functioning of the electronics.
- Single electronic components today cannot be made with a reliability satisfying our needs.

Therefore we have to:

- Detect any occuring failure
- Isolate a failure and undertake appropriate actions such that the functioning of the system is either not affected or is reduced to a predetermined permissible level
- Indicate any occurring failure

3. STANDPOINT AND HISTORICAL BACKGROUND

3.1 Standpoint

Statement I:

It is impossible to detect a malfunction without extra effort which exceeds that necessary to merely fulfil the function.

Statement II:

The additional effort to detect and possibly isolate a malfunction regardless in which packaging it is sold, (monitoring, paritybit, selfdetection, skewing, etc.) means redundancy.

However, there are many ways of detecting a failure and they differ greatly from one another in terms of effort and applicability.

Statement III:

In order to detect any failure only two basic principles apply:

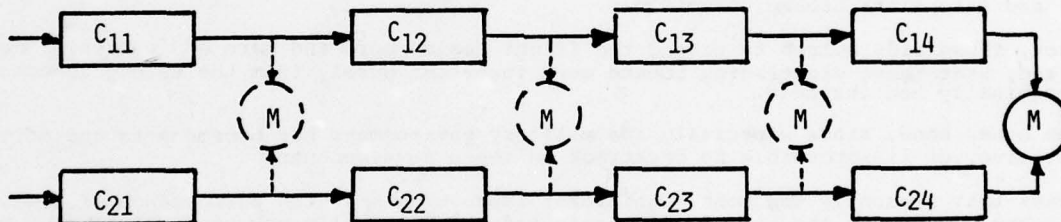
Discrepancy and Majority decision

3.2 Historical Background

In order to understand the authors view and approach a very brief step back into well known solutions has to be presented.

3.2.1 The Duplex System

As well known, two channels of the same kind may be used to compare their signals.



As simple as this seems to be, it opens up major problems:

- a. In case of a failure it is not possible to decide which channel has failed.
- b. It is not possible to prevent failure propagation (blocking).
- c. If any one component of one channel fails, one whole system fails.
- d. Therefore only the final monitor/comparator (M₃) makes sense.
- e. The failure detection depends only on the comparator/monitor at the end of the channels. Therefore it has to be more reliable and at least as redundant as the system channels.
- f. The system design principle, because there are no other criteria, has to be: In case of doubt, failure. Therefore this system has a higher tendency to degrade, to indicate a failure than to survive.

In conclusion it has to be stated, that if not used in combination with other methods, a duplex system is very ineffective and inferior to all the other principles discussed here.

The generalisation of a signal comparison which indicated only, "there is a discrepancy", but does not allow the determination of which component or channel caused it, will be called

"Discrepancy Principle".

A discrepancy therefore allows only the conclusion:

"There is a failure".

This occurs in many different applications such as the previous two channel system, parity bit, failure self detection, observer technique etc.

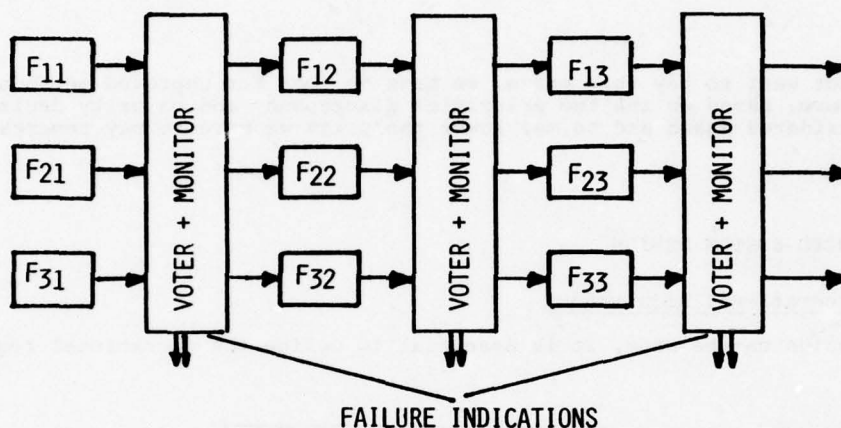
These all have the same characteristics:

A failure occurring can be detected and indicated but it will cause the automatic exclusion of both channels, from further use since without additional measures (i.e. a third channel) a failure localisation and blocking is impossible. This, of course, means that its use alone in a flight safety critical position is not permissible.

3.2.2 The Multi-Channel Parallel System

Obviously, the two channel approach is not promising but it was useful to explain general problems and prepare solutions. Two channel systems or the application of the discrepancy principle, however, may be found as sub-systems of a multi channel system. Multi channel systems work based on the statistical principle that the probability of occurrence of a simultaneous multiple failure is general an order of magnitude smaller than of a single failure.

Consider the example of a triplex system:



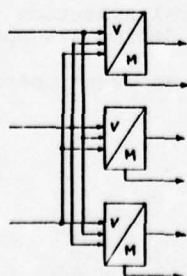
It has the following characteristics:

- A first failure, by using majority decision, can not only be detected but also localised.
- Dividing the system into blocks allows a signal consolidation after each block which in turn allows the system to sustain as many first failures as blocks are present.
- After one first failure in a block the system can work without degradation in function and performance. However, any further (second) failure in one block can only be detected but not survived.
- In case of a simultaneous double failure (in one block) it is not able to detect and survive it. However, while for a two channel system the problem is valid for the whole system (=many components) it here is restricted to the components within one block (=fewer components).

Generalised, a N-channel parallel redundant system can survive $N-2$ non-simultaneous failures and can detect one more ($N-1$); ($N \geq 3$).

The key component in such a multi-channel system is the voter. The voter isolates the failure to one block and consolidates the signals such that behind the voter they are all alike. Any user behind the voter does not notice that there was a failure before it.

On the other hand, each voter is not just one piece, it has to be redundant as the system is. For example in a 3-channel system the voter has to be triplicated for each block. With that, the failure detection and isolation would be solved, but there is a price:



Voters and Monitors in Redundant Systems have to be redundant!

A multi channel system consisting of

n channels and

m blocks

can sustain $n - 2$ non-simultaneous failures

can indicate $n - 1$ non-simultaneous failures

If no special measures are undertaken, the effort for such system grows extensively

The number of functional components increases linearly with the number of channels

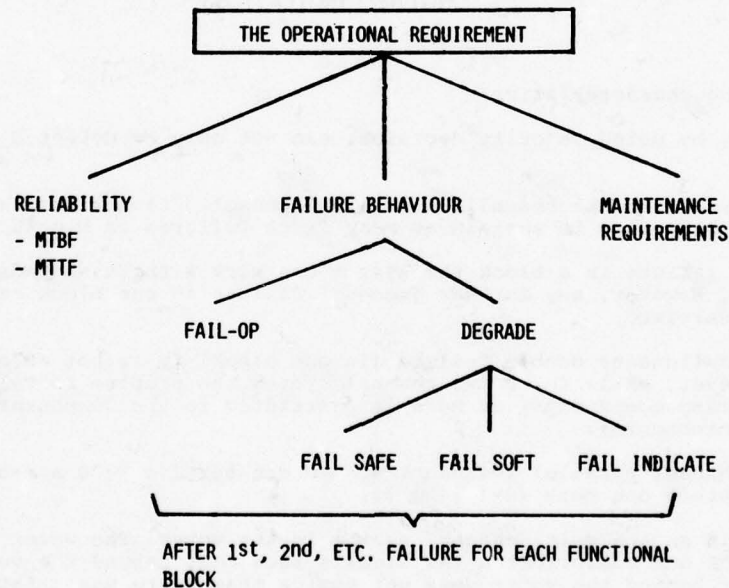
The number of additional voters and monitors increases as $m \times n$

Since we do not want to pay this price, we have to look for unproved methods. They are, as stated before, based on the two principles discrepancy and majority decision but will better be considered again and so may lower the price we have to pay remarkably.

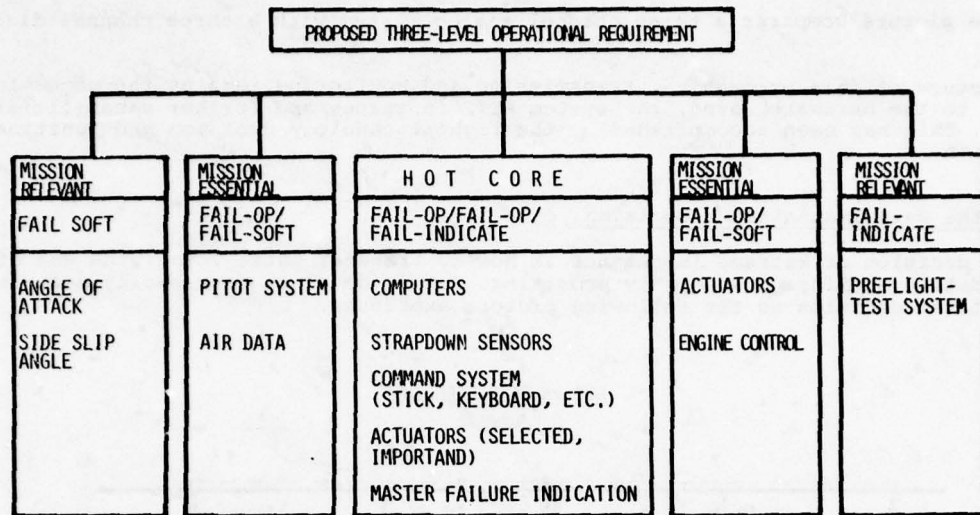
4. ADVANCED SYSTEM DESIGN

4.1 The Operational Requirement

Before any design can be made, it is essential to define the operational requirement:



In this paper we will restrict ourselves to the technical assertion "failure behaviour". It is of course the honour and duty of the customer to define his operational requirement, but he may want our advice. For a fighter/combat aircraft here is a proposal for a graduate operational requirement for a guidance and control system.



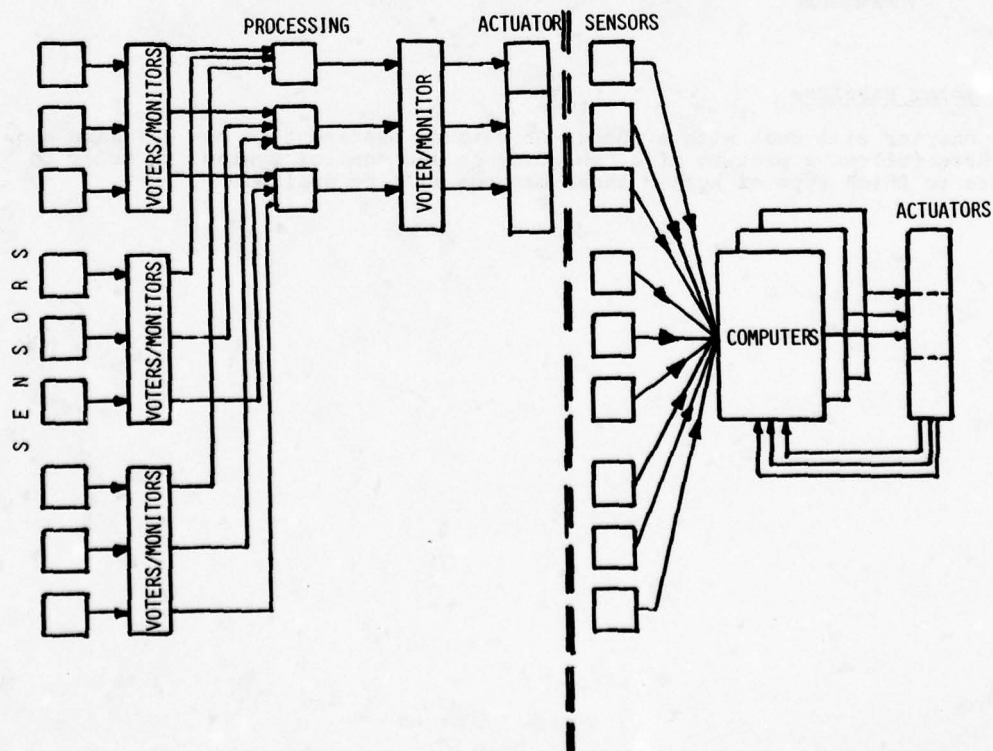
4.2 The Technology Decision

Obviously, the failure rate of a system is a function of the individual component reliability and the number of components. The first decision (which is both mandatory and beneficial) is to go digital.

With the advent of digital data processing and transmission the transition from continuous computation and transmission (= single purpose use) to multiplexed systems took place. Computers and data transmission lines are not needed on a continuous basis. If the computer speed and data transmission rate has been selected high enough the available time can be utilized for different tasks.

Without aggravating the classic block structure the same functions as before are performed but the hardware in order to do this is drastically reduced.

STRUCTURE OF AN ANALOG vs A MULTIFUNCTIONAL DIGITAL SYSTEM



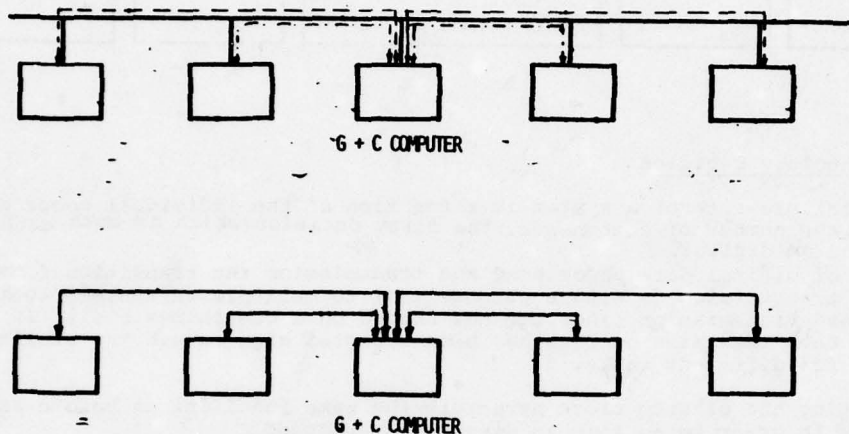
The above picture compares a three channel analog system with a three channel digital system.

The structure of data processing, transmission and monitoring remains the same. In addition to the hardware saved, the system MTTF increases and further capabilities are obtained. This has been accomplished by the right technology decision and functional integration.

4.3 The Data Transmission Decision

The next decision of extreme importance is how to transfer data. Today, the use of a digital data transmission is highly promising. But it does not necessarily mean an architectural decision as the following picture expresses.

BUS vs STAR STRUCTURE

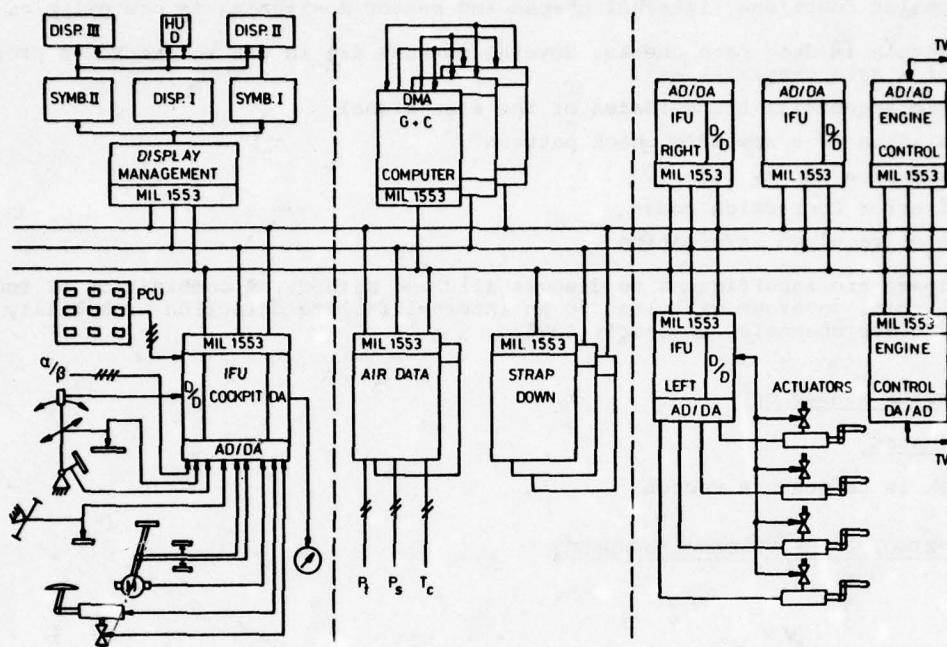


1. INDEPENDENTLY FROM BUS OR STAR ARCHITECTURE MOST INFORMATION WILL GO VIA THE G + C COMPUTER
2. THE GREAT PROGRESS LIES IN THE USE OF STANDARDIZED DIGITAL SERIAL DATA TRANSMISSION

4.4 System Baseline

The next chapter will deal with a number of methods applied to a new guidance and control system. Here follows a picture of a new guidance and control system, in order to illustrate to which type of system these methods will be applied.

STRUCTURE OF A DATA BUS ORIENTED GUIDANCE + CONTROL SYSTEM

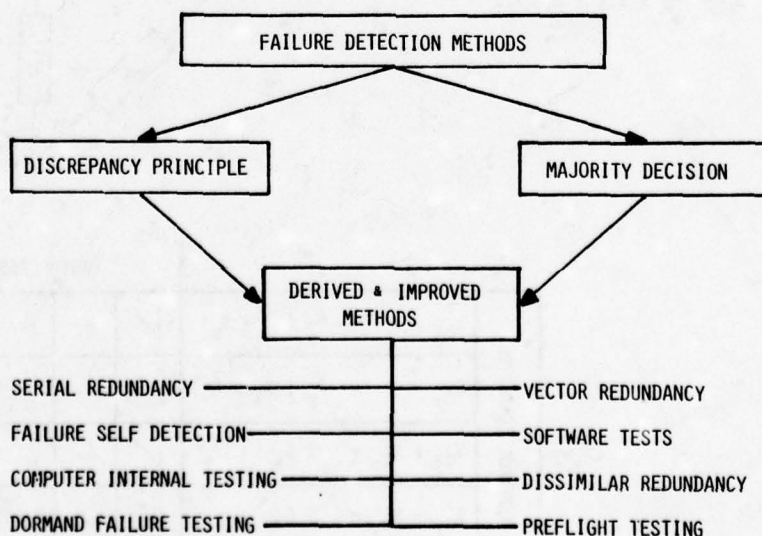


This g & c system will be fail-op/fail-op utilizing majority decision, discrepancy principles and the improved methods of the following chapters.

5. IMPROVED FAILURE DETECTION METHODS

5.1 Outlook

The use of digital technology and computer programs has lead to a number of new methods which despite the fact that they are all based on majority decision and the discrepancy principle, can save hardware considerably.



The following paragraphs are dedicated to some selected examples.

5.2 Serial Redundancy

Serial redundancy means, that the same component may be used for different functions. The multi-functional use of the guidance and control computers for the computation of control laws, autopilot functions, internal checks and sensor monitoring is one example.

Another example is data path checks. Several methods are in use to check the proper function of a data transmission

- multiple sequential transmission of the same signal
- transmission of a specific check pattern
- addition of a parity bit
- use of error correction codes
- return of received information

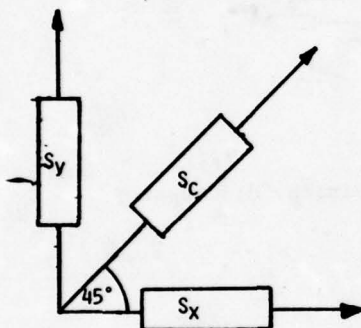
Time and space are insufficient to discuss all these methods. A combination of some multiple partial coverage will lead to an internal failure detection probability (without the aid of other channels) of nearly 100%.

5.3 Vector Redundancy

5.3.1 Sensors

If the task is to measure vector

BASIC PRINCIPLE OF VECTOR REDUNDANCY

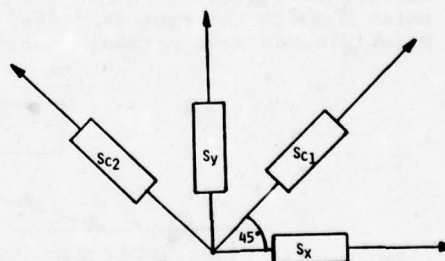


MONITOR SENSOR

MONITOR EQUATION

$$S_c \stackrel{!}{=} \frac{1}{\sqrt{2}} (S_x + S_y)$$

FAIL-OP/FAIL INDICATE ARRANGEMENT OF FOUR SENSORS IN ONE PLANE

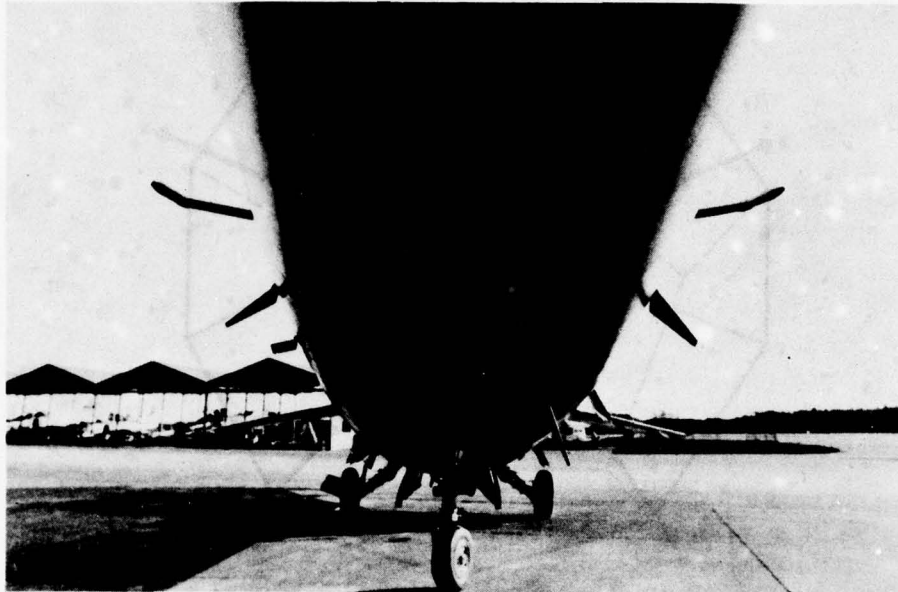


		TRUTH TABLE				
CONTROL EQUATION	$S_{c1} = \frac{1}{2} (S_x + S_y)$	✓	-	-	-	✓
	$S_{c2} = \frac{1}{2} (-S_x + S_y)$	✓	-	-	✓	-
	$S_{c1} + S_{c2} = 2 S_y$	✓	-	✓	-	-
	$S_{c1} - S_{c2} = 2 S_x$	✓	✓	-	-	-
		ALL OK	Sy defect	Sx defect	Sc1 defect	Sc2 defect

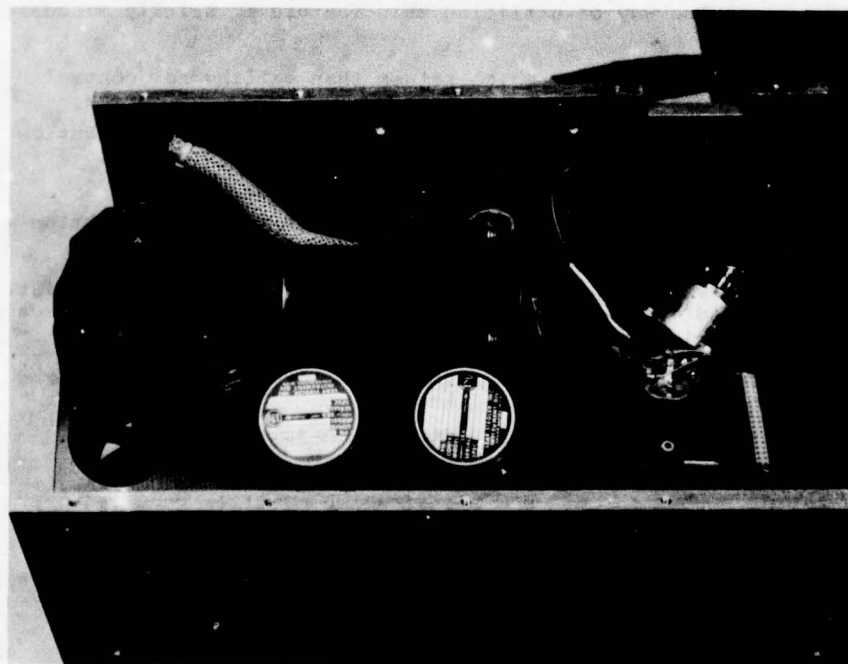
like in the left picture. One can add a control sensor which, by using the discrepancy principle in conjunction with the other signals, determines if one out of the three is failing.

In the right picture the principle is extended and we can show with a few control equations that the result is fail-up/fail indicate behaviour.

Compared to parallel redundancy a significant amount of hardware can be saved. However, it basically works with majority decisions and therefore, is a derivative of parallel redundancy.



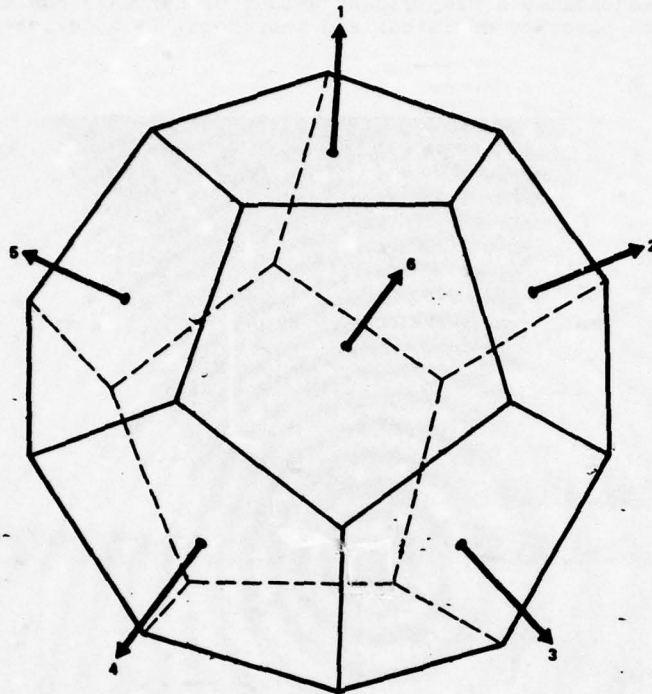
Vector redundancy applied to the measurement of the angle of attack and sideslip in an aircraft



Monitor sensors in a vectangular arrangement of rate gyros and accelerometers

The principle has already been successfully applied. The sensor orientation can be chosen such that sensitivity problems are solved simultaneously.

Remark : It goes without saying, that as long as no sensor fails, all may be used in order to improve the total sensor system accuracy.



(Fail-op)³/fail indicate arrangement of single axis sensors using vector redundancy

5.3.2 Control Surfaces

For aerodynamic, structural or other reasons, especially in new designs, more control surfaces are available than degrees of freedoms to be controlled. This is called an overdetermined system. One way of utilizing this feature is briefly discussed in the following :

- Step 1) : Compute a control law which rather than calling for control surface movements, asks for control momentums and forces.
- Step 2) : Send the required forces and momentums through a distribution matrix which optimally utilizes the available controls.
- Step 3) : Monitor the proper function of your control
- Step 4) : In case of a control surface failure have a new distribution matrix ready which takes that failure into account.

Step 4 causes the problem. For any first failure there are as many new distribution matrices required as individual controls can fail. They have to be precomputed and stored in the system. This seems to be feasible.

But in case of a second failure a permutation of failures has to be covered. If m is the number of controls it looks like :

first failure :	m	new distribution matrices required
second "	m^2	new distribution matrices required
third "	m^3	new distribution matrices required

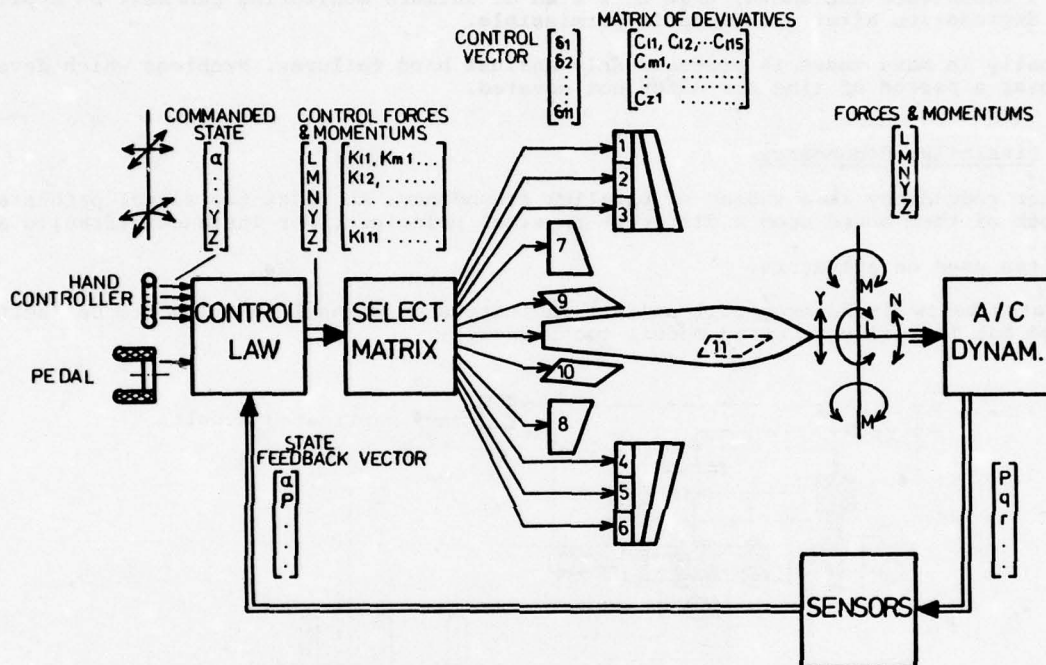
After a second failure the method is questioned.

Additionally this way is limited to that point where the remaining controls form a determined system to control the desired degrees of freedom.

In case of any further failure, the system degrades. That means here that not all original degrees of freedom can be controlled independently of each other and may progress to the point where the airplane is not controllable at all.

It should also be mentioned, that after failures below a determined system point, changes in the control law are likely to be necessary.

CONTROL VECTOR REDUNDANCY



5.4 Failure "Self" Detection

Failure self-detection as such does not exist. That, what is called "self" detection is a combination of internal parallel or serial redundancies and discrepancy principles within the device.

Despite of this, the methods called so can be extremely useful. Discussing them even partially costs too much time and space. Some are listed below :

- test of all supply voltages, pressures or other energizings
- plausibility tests, auxiliary measurements
- spin frequency tests of gyros
- pre-known test voltages among signals in an A/D converter
- wrapping around any output signal and testing its correct re-appearance
- watch dog timing
- internal error correction codes
- partly parallel redundancy like dual CPU's in a computer
- instruction set testing
- superposition of test signals

Obviously they all need an excellent knowledge about the components and the internal process to be monitored and can be developed in combination of each other.

Suppliers state to achieve a 95 - 98% certain average of self test for components such as IMU's.

5.5 Software Tests

From the methods which may be realized in software, filter techniques have been taken as an example.

Filter techniques, in general mechanized in software, can be used to determine a malfunction.

Therefore, it should be noted that in most cases where filter techniques are applicable, some kind of inherent redundancy does exist already. Via filter techniques the following variables may be monitored against each other

rates	↔	attitudes
velocity (aerodyn)	↔	internal velocity
α, β	↔	α_{in}, β_{in}

Practical experience has shown, that this kind of failure monitoring can best be applied if some degradation after a failure is permissible.

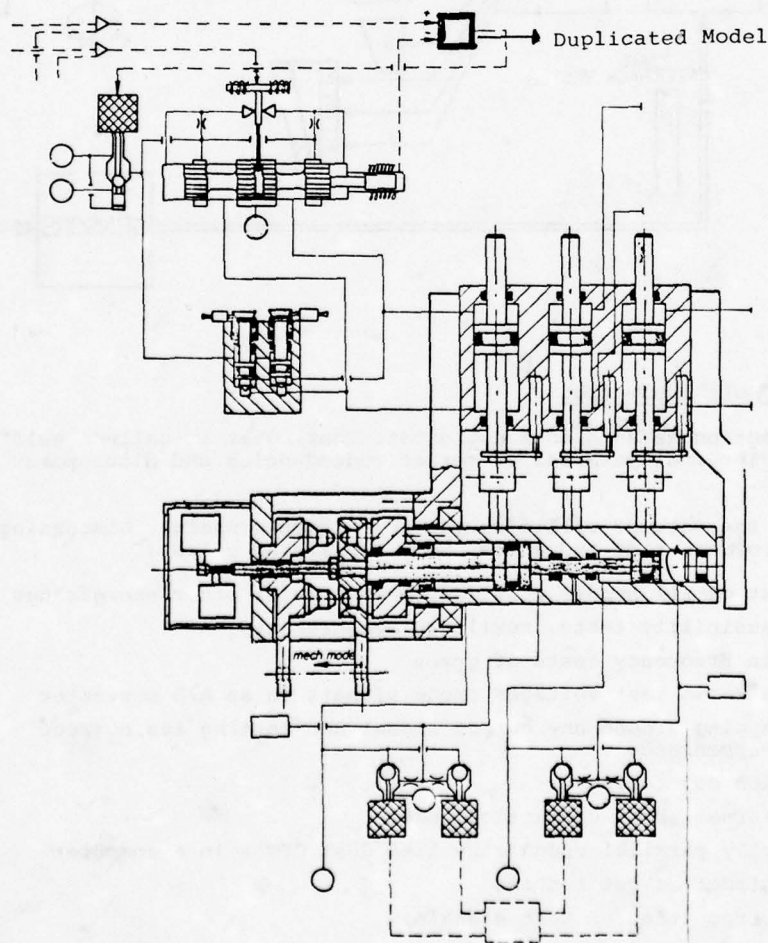
Additionally in many cases it protects only against hard failures. Problems which develop slowly over a period of time are often not covered.

5.6 Dissimilar Redundancy

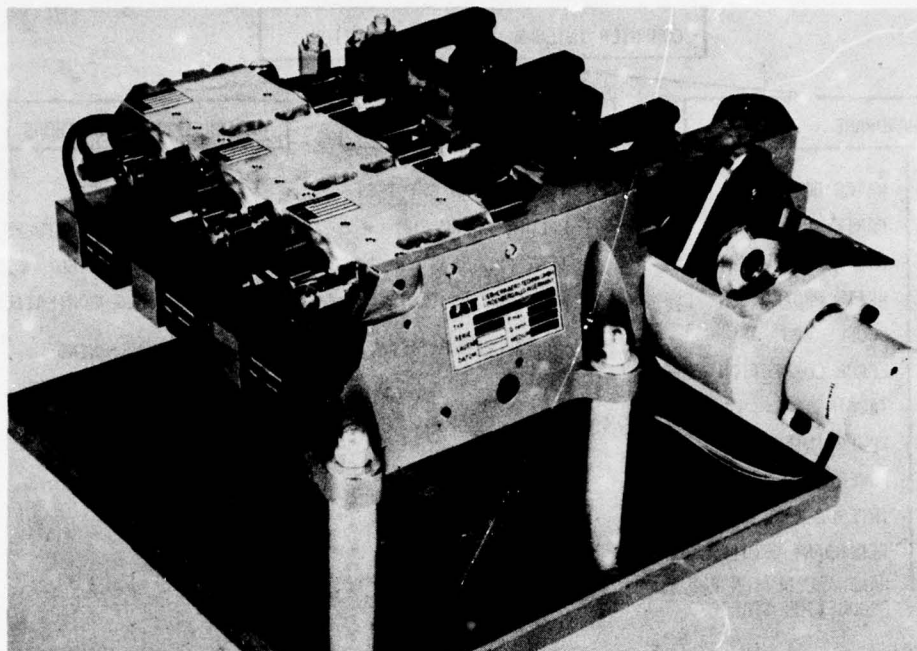
Dissimilar redundancy is a subset of parallel redundancy. At least two signal pathes are used, both of them based upon a different physical principle. For instance hydraulic and electric.

It is often used on actuators.

The actuator below is fail-op/fail-op/fail indicate each channel is called to be "self" monitored but it really is using modell techniques.



Blockdiagram of a failure "self" monitored actuator



Three channel (fail-op, fail-op, fail-indicate) actuator

The behaviour of the hydraulic part of one channel is electrically modelled and compared in a monitor. In case of discrepancy this channel is hydraulically by-passed. Because of failure considerations it turns out that the electrical part of each channel has to be duplicated, so the monitor.

What one finally got and what looks like a three channel system was a system consisting of three independent sub-systems (channels), each of which consisting of two electrical and one hydraulic channel. A total of nine channels. It still was a great progress because it

- saved one hydraulic channel
- avoided hydraulic cross monitoring

The additional effort has been directed towards those parts which were

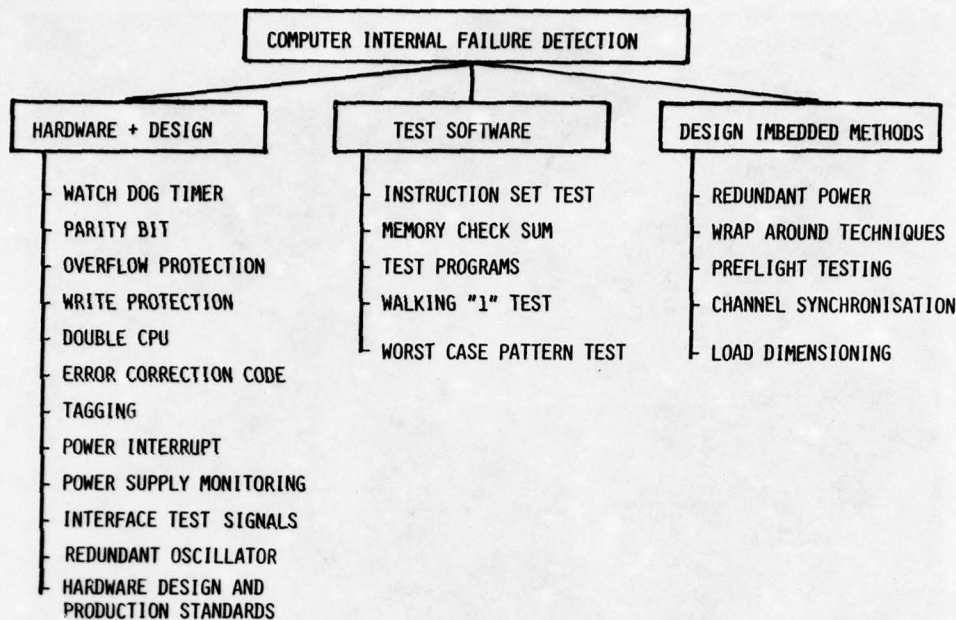
- more reliable
- smaller
- easier to test
- less expensive

On the other hand even this example showed the limitations of dissimilar redundancy : The final decision were made within one privileged system, because it knew more about the other system than vice-versa.

Furthermore it was difficult to model the other, dissimilar (hydraulic) channel. A problem which becomes much more severe in long process chains.

5.7 Computer Internal Redundancy and Failure Detection

Since the computers in modern G + C systems are the most important multifunctionally used devices, the methods which are available to achieve internal failure detection capability are summarized here. Many of them have been mentioned elsewhere, some are new.



5.8 Dormant Failures

Important components of a guidance and control system may be used only in certain cases. So for instance most components which get activated only in case of a failure. But then they have to work! In case of such components which cannot be tested during flight, we depend only on pre-flight testing, reliability figures and redundancy.

Some possibilities for the prevention of dormant failures are given

- test signals
- superimposed signals
- off-set

5.9 Pre-flight Testing

"Never take off without a complete failure free system"!!!

Pre-flight testing is the only possibility to test all components, even such which are activated only in case of a failure.

In order to avoid dormant failures, special tests have to be performed, far exceeding those tests which take place during normal operation.

These general principles should be observed :

- while pre-flight testing, normally also use and validate the inflight tests
- simulate any possible failure and combinations of failures
- test for proper failure isolation and indication
- activate all components, especially such which are not used during normal operation
- test signal generation must cover the whole range of measurement, displacement, data, calibration curves etc. Check for proper computation. This, at least pointwise.
- keep the error source "human being" as small as possible

No doubt, this in many instances means additional effort in hardware and software over and above that needed for the performance of the nominal function.

The tendency therefore, is to save on the test installation.

This is permissible. But :

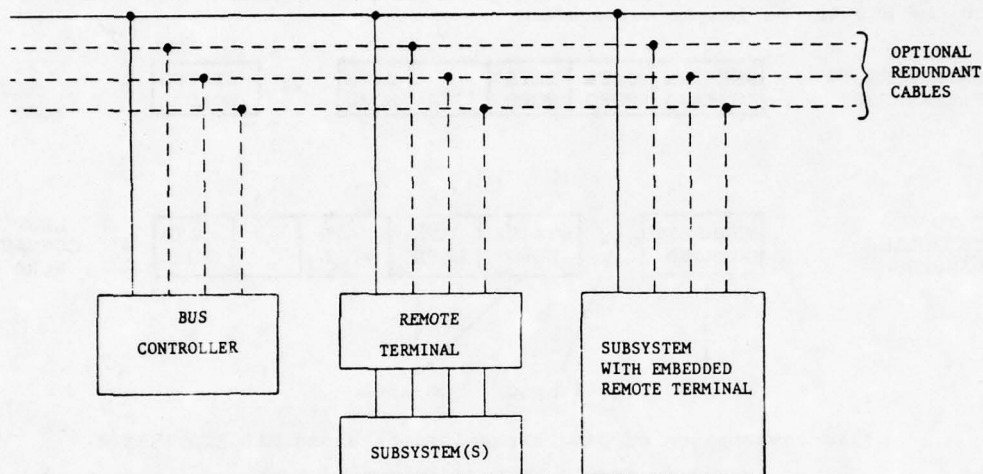
- Test conductions must not be affected and clearly provable
- Test initialisations have to be reliable and at least as redundant as the system to be tested. An arbitrary (i.e. inflight) pre-flight test initialisation would be deadly.

It goes without saying, that the flight safety aspects of the pre-flight testing normally will be combined with the maintenance aspects. If possible, one wants to know the very exact location of a failure for equipment replacement and repair. In addition a "condition" of the equipment may be determined to allow "on condition maintenance".

6. DATA BUS ORIENTED GUIDANCE AND CONTROL SYSTEMS

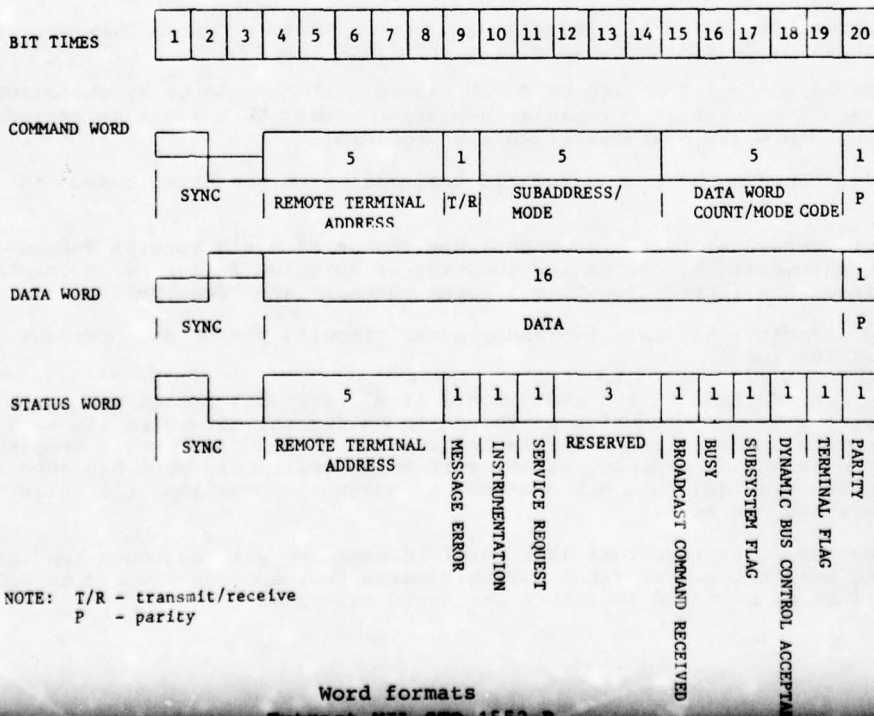
6.1 Initial Offers by MIL STD 1553 B

At the first glance, MIL STD 1553 B seems to have a heart for redundant sufferers. It refers to redundant applications.



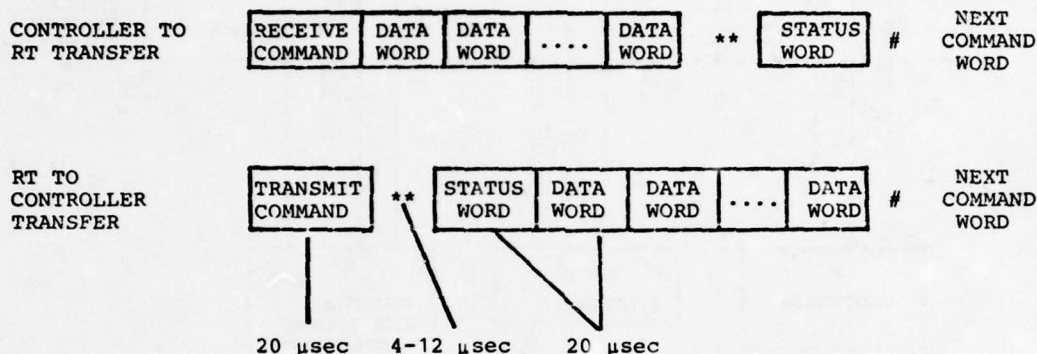
Sample multiplex data bus architecture
Extract MIL STD 1553 B

At the second glance, this was almost all, what MIL STD 1553 B had to offer for redundant applications. However, some of its features help.



- a. Word sync: Helps to synchronize redundant channels and helps to prevent error propagation over more than one word.
- b. Status word: Assists in-line monitoring, but the deficiency is, that no information is given about what happened while sending.
- c. Parity bit: In serial applications of rather limited value.
- d. Data rate: 1 Mbit is sufficient for a sensor to computer to actuator data transmission. Because of the bus conventions it hardly exceeds 30,000 words per second of effective data rate. Not suited for a computer to computer communication which is needed for fast failure detection and isolation.

Special care has to be undertaken to avoid intolerable transport lags between sensors, computation and actuation. In many cases a synchronisation of the devices to certain transmission times and a well considered bus protocol are required. This means, that the devices on the bus are no longer autonomous!



Time consumption of data transmission, using MIL STD 1553 B

6.2 A Redundant Design, Using MIL STD 1553 B

As stated in chapter 4.1 there is a (fail-op)² requirement assumed for the hot core of the guidance and control system. The solution here is a three channel failure self monitoring system.

The following steps have been undertaken:

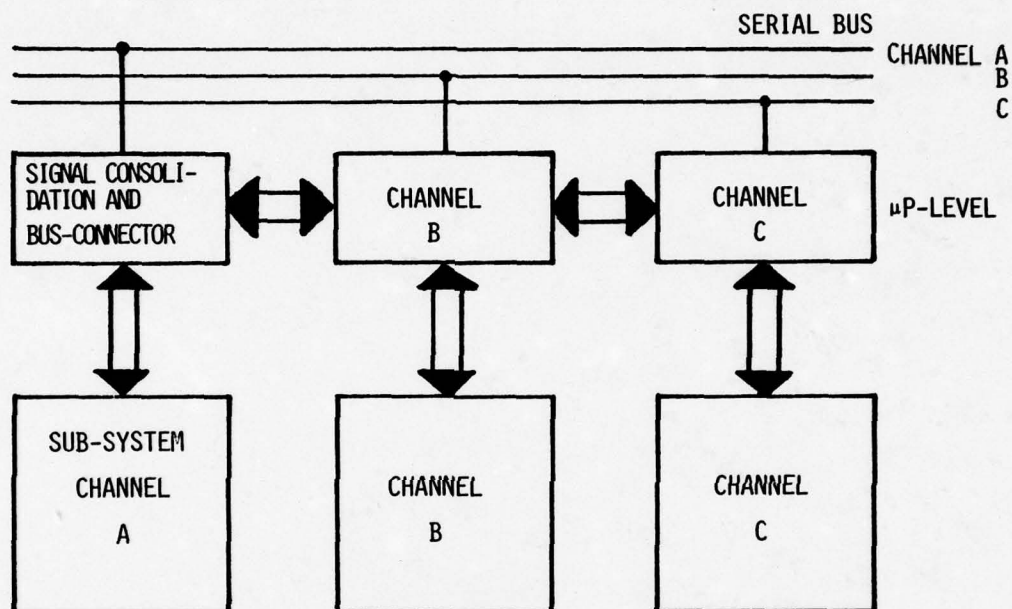
- a. Synchronisation: Between all three channels to be a maximum of 5 μsec. This eases the failure detection, decreases especially the failure thresholds.
- b. Device synchronisation: Software or device (Terminal) have to be synchronized to have the data ready and fresh at certain transmission times. This again decreases failure threshold and decreases transportation lag problems.
- c. Consolidation: Sending of unconsolidated information on the three busses is not permitted.
- d. First failure coverage: Each (redundant) device has to use a special redundant interface, which automatically blocks any incoming or outgoing failure such that behind that interfaces the information on all three channels are identical.
- e. Antijamming circuits: At least two independent circuits per channel prevent jamming in each businterface.
- f. Life status: In addition to the status word of MIL STD 1553 B each message will be concluded with status information contained in an (extra) data word (in this way MIL 1553 formats are maintained). This information is generated while sending the message. Other than the ordinary status word which tells only what happened before the message was started, this one contains important information of a failure occurred while the message was sent.
- g. Error correction code: Important (hot core) information will be converted into ECC-format prior to sending. Unfortunately this means that some data words in ECC-format do not fit into 16 bits and therefore use two data words.

h. Second failure coverage: Will be achieved by

- discrepancy indication
- + failure self detection by ECC
- + message resent (serial redundancy)

i. System reconfiguration: Has been considered, however, no final decision has been made yet. The problem is the handling of the interwoven complicated logics.

j. DMA: A fast DMA connection between the main G & C computers will enable the computers to react fast upon any failure.
Superior decision will remain by the main G & C computers. Therefore the DMA is considered mandatory.



Principle of Redundant Databusinterface

7. CONCLUSIONS

The spectrum of new techniques for failure detection and isolation has been considerably widened. This will result in less hardware effort with improved result.

On the other hand it requires a tremendously growing know how at the level of system designers and integraters.

This has to be complemented by similar capabilities and effort at the component, sensor, devices, equipment designer, and manufacturer.

The fight against the malfunction can be fought best during design and beginning at the lowest level.

L'INTEGRITE DES LOGICIELS EMBARQUES : UNE SOLUTION

G. GERMAIN

ELECTRONIQUE MARCEL DASSAULT

55, quai Carnot

92214 - SAINT CLOUD

FRANCE

RESUME :

On présente une solution destinée à assurer l'intégrité du logiciel embarqué sur avions et engins. Le but principal est d'accroître la sécurité du logiciel afin de la rendre homogène avec celle du matériel, qui, dans le cas considéré, est très élevée. On obtient des retombées intéressantes au niveau de la fiabilité et de la maintenabilité du logiciel ainsi que des coûts de validation. Les moyens mis en oeuvre ont été conçus afin d'être les plus simples et les plus économiques possibles. Ils s'appuient sur une structure du logiciel et une machine virtuelle du calculateur bien adaptées aux applications embarquées. L'accent est mis sur les mécanismes de contrôle de l'adressage qui permettent d'éviter toute destruction intempestive du logiciel.

1. INTRODUCTION

La numérisation des fonctions à bord des avions et des engins s'est rapidement développée au cours des dernières années, et elle est appelée à croître encore dans la prochaine période.

Cette évolution est commandée par le caractère de plus en plus complexe des missions et elle est rendue possible par les progrès de la technologie des composants et l'accroissement important de la puissance de calcul qui en résulte, pour un volume et une consommation donnés.

Ainsi, le sous-système informatique, plus ou moins complexe selon le cas, se voit attribuer une autorité grandissante et dès à présent déterminante dans l'accomplissement des missions.

Ceci pose le problème de la sûreté de fonctionnement de ce sous-système, face aux exigences opérationnelles de sécurité, de fiabilité, de disponibilité et de maintenabilité.

Deux aspects importants de ce problème sont les suivants :

- garantir une sécurité et une fiabilité intrinsèques élevées pour les fonctions ayant un caractère vital ou critique dans la mission. Tant qu'une telle garantie ne peut pas être donnée, des fonctions comme le pilotage automatique des avions dans certains domaines de vol ne peuvent pas être envisagées.
- préserver les qualités intrinsèques ci-dessus en assurant l'intégrité de ces fonctions face aux agressions de leur environnement ou aux défaillances de leur support matériel. En particulier, il faut pouvoir protéger chaque fonction vitale contre les défauts de fonctionnement d'autres fonctions qui cohabitent avec elles dans la même mémoire, par exemple. Ce besoin d'intégrité est également motivé par la nécessité que l'intégration de fonctions placées sous la responsabilité de différents contractants se fasse dans des conditions acceptables pour chacun d'entre eux et pour le maître d'oeuvre.

Ce problème se pose aussi bien quand les fonctions sont distribuées, réparties dans différents microprocesseurs d'équipements, que lorsqu'elles sont intégrées, centralisées dans un calculateur principal. Dans ce dernier cas, il se pose cependant avec plus d'acuité car les différentes fonctions sont situées dans la même mémoire, partagent les mêmes ressources de calcul et d'entrées-sorties, et leurs interfaces sont donc moins bien matérialisées que dans le cas d'un système distribué. C'est surtout à l'intégrité et à la sécurité des fonctions centralisées dans un calculateur principal que nous nous intéressons ici.

La perspective que nous considérons est celle des missions de la prochaine décennie. Si actuellement on peut obtenir une sûreté de fonctionnement suffisante au regard des exigences opérationnelles avec des structures classiques du matériel et du logiciel, cela ne sera plus possible dans l'avenir. Il sera alors nécessaire d'intégrer dans le logiciel comme dans le matériel des fonctions de détection et de recouvrement des erreurs pouvant seules permettre de réaliser des systèmes fiables avec des composants moins fiables.

Le calculateur COPRA - qui fait ici même l'objet d'une autre communication (1) - est conçu pour satisfaire à ces exigences futures. Son architecture ainsi que ses mécanismes de détection, de reprise et de recouvrement lui confèrent une tolérance élevée vis-à-vis des pannes permanentes et transitoires du matériel ainsi que de certaines perturbations externes. Ces caractéristiques sont décrites dans la communication précitée, et il n'est pas utile d'y revenir ici. Elles permettent d'atteindre une fiabilité et une sécurité du matériel très élevées, de plusieurs ordres de grandeurs supérieures à celles d'une structure non redondante basée sur la même technologie. De plus, et c'est là un point fondamental, elles interdisent,

par principe même, toute destruction incontrôlée du logiciel à la suite de pannes du matériel ou de perturbations de l'environnement. Ainsi, l'intégrité du logiciel ne peut être mise en cause que par les propres erreurs de celui-ci.

Il reste donc à résoudre ce problème des erreurs du logiciel, car il ne sert à rien de disposer d'un matériel ultra-fiable si la fiabilité du logiciel n'est pas d'un niveau comparable.

C'est à la présentation générale d'une solution de ce problème, développée dans le cadre du projet COPRA, qu'est consacrée la suite de ce document.

2. FIABILITE DU LOGICIEL :

La fiabilité du logiciel dépend de deux facteurs :

- sa correction, qui correspond au nombre et à la "gravité" des erreurs qu'il contient à l'instant 0 de la mission. Ces erreurs peuvent avoir leur origine dans l'analyse fonctionnelle (mauvaise définition de ce que doit faire le logiciel) ou dans la programmation (mauvaise réalisation). Elles peuvent également être introduites à l'occasion de la maintenance du logiciel.
- sa robustesse, qui caractérise la capacité du logiciel à tolérer en cours de mission, soit la manifestation de ses propres erreurs résiduelles, soit des valeurs d'entrée ou des actions d'opérateur erronées, soit des conséquences de pannes du matériel ou de parasites. (Pour COPRA, ce dernier point est sans objet, comme on l'a indiqué précédemment).

La robustesse du logiciel dépend en grande partie de sa capacité à conserver son intégrité, c'est-à-dire à ne pas être détruit ou modifié intempestivement par les événements précédents.

2.1. Correction :

La correction du logiciel peut être approchée par deux moyens complémentaires :

- Réduction des causes d'erreurs : il s'agit d'améliorer la qualité des travaux de développement du logiciel en agissant simultanément sur trois fronts : méthodologie, technologie, moyens humains.

Les dix dernières années ont vu des progrès importants dans les domaines de la méthodologie et de la technologie : prise de conscience de la nécessité d'une méthodologie rigoureuse de développement du logiciel et reconnaissance de l'importance fondamentale de la phase de définition ; apparition et développement des techniques de programmation structurée, de langages et d'outils d'aide à la conception et de langages de programmation évolués et adaptés aux applications en temps réel.

Ces progrès ont permis certainement de mieux maîtriser le processus de production de logiciels, et donc de réduire les causes d'erreurs. Cependant, le travail de développement de logiciel reste un travail complexe, peu automatisé, et donc sujet à l'erreur humaine.

- Recherche et réparation des erreurs : il s'agit de détecter et de corriger le maximum d'erreurs avant la mise en exploitation du logiciel.

La technique la plus utilisée reste le test avec ses deux aspects : vérification de la logique interne des programmes (ou tests statiques) et validation des programmes dans leur environnement réel ou simulé (tests dynamiques).

Les techniques d'analyse des programmes - avec leurs différentes variantes : relecture de code, inspections, "walkthroughs" - sont de plus en plus fréquemment utilisées.

Des techniques nouvelles telles que l'exécution symbolique ou les preuves formelles de programmes sont loin d'être opérationnelles.

Le point important, quelles que soient les techniques utilisées, est qu'aucune ne permet de garantir l'élimination de toutes les erreurs. Les tests et les analyses permettent seulement d'établir la présence d'erreurs, mais ne prouvent jamais leur absence. L'expérience montre cependant que plus un logiciel est vérifié et moins il contient d'erreurs : c'est pourquoi, dans l'état actuel de l'art, le test est le principal moyen utilisé pour accroître la fiabilité du logiciel, et une part importante du budget de développement y est consacrée lorsqu'on recherche une fiabilité élevée.

2.2. Robustesse :

La recherche de la correction comme seul moyen d'accroître la fiabilité et la sécurité, expose à deux inconvénients majeurs :

- On ne peut jamais démontrer que la correction est absolue (l'expérience tend plutôt à démontrer le contraire), ni même quel est le degré de correction atteint.

- Même s'il ne reste qu'une erreur dans le programme, on ne connaît pas la probabilité pour que cette erreur se manifeste ou non pendant la mission, dans les 5 premières ou les 5 dernières minutes, en détruisant la moitié du contenu de la mémoire ou en n'ayant que des conséquences mineures. Ceci est particulièrement le cas pour des logiciels de contrôle en temps réel comme les logiciels embarqués. En effet, pour ce type de logiciels, aussi poussée que soit la validation, il est matériellement impossible de tester toutes les configurations opérationnelles potentielles, ni même une partie significative de ces configurations.

Une erreur résiduelle, par définition, ne se manifeste en cours de mission que lorsqu'apparaît une telle configuration non testée, et il est très difficile de faire des hypothèses sur la probabilité d'apparition de ces configurations ou sur l'instant de leur occurrence. Très difficile également de prévoir les conséquences d'une erreur inconnue.

C'est ce fait qui, avant tout autre, motive l'intégration au matériel et au logiciel opérationnels de fonctions de détection, de confinement et éventuellement de recouvrement des erreurs logicielles.

Il peut être envisagé deux degrés d'implémentation de telles fonctions de tolérance aux erreurs :

- détection et confinement (ou non propagation) seuls si l'on cherche uniquement à accroître la sécurité. C'est le cas pour la plupart des logiciels embarqués actuels où le système est capable en cas de défaillance signalée du logiciel, de continuer à assurer une mission réduite en raison de redondances existant au niveau système pour chaque fonction critique du logiciel.

- recouvrement des erreurs si on cherche également à accroître la fiabilité. Ce sera le cas pour les logiciels avioniques futurs dont certaines fonctions devront pouvoir survivre même si elles sont le siège d'erreurs.

2.2.1. Détection :

La détection des erreurs résiduelles du logiciel pendant son exploitation implique l'adjonction de mécanismes aux niveaux matériel et logiciel. Ces mécanismes doivent assurer une détection suffisamment rapide pour éviter toute propagation dangereuse des erreurs. En particulier, ils doivent garantir la non-destruction du logiciel, c'est-à-dire son intégrité.

Trois types principaux de mécanismes doivent être mis en oeuvre :

- Contrôle des sorties (données ou commandes) par des tests logiciels de vraisemblance permettant de vérifier que certaines assertions sur les valeurs des données de sortie ou sur les commandes sont respectées. Ces tests sont spécifiques de chaque application et ne peuvent être mis en oeuvre qu'au niveau du logiciel d'application. Ils sont indispensables et d'ailleurs assez fréquemment utilisés dans les logiciels embarqués. Mais ils ne suffisent pas, à eux seuls, à assurer un taux de détection suffisant.

- Contrôle de l'exécution et des durées. Faisant toujours appel à un mécanisme du type "watchdog" basé sur l'utilisation d'une horloge temps réel, ils permettent de détecter les blocages de certains processus, à la suite d'erreurs de synchronisation ou de la monopolisation de certaines ressources par un processus (erreur sur test de fin de boucle, par exemple). Ils doivent être implémentés au niveau d'un moniteur temps réel qui a seul les informations nécessaires au contrôle de l'exécution des tâches.

- Contrôle des droits d'accès et d'utilisation : c'est le mécanisme de détection fondamental qui conditionne l'efficacité des deux précédents. Schématiquement, il consiste à garantir qu'un processus ne peut pas faire autre chose que ce qu'il a à faire, même s'il le fait mal. En particulier, il permet de s'assurer que les contrôles sur les sorties ne peuvent pas être court-circuités par une erreur d'adressage, de même que les contrôles des durées. Le principe général est de vérifier que chaque processus n'adresse pas d'autres objets que ceux auxquels il a droit et qu'il ne peut en aucun cas exécuter d'instruction qui lui assurerait des droits supplémentaires. De tels mécanismes ont reçu diverses formes d'implémentation : contrôle de l'adressage par protection d'une mémoire segmentée ou paginée, modes maître-esclave autorisant ou interdisant l'utilisation de certaines instructions, machines à capacité ou à domaines (2, 3, 4) etc... La solution présentée ci-après concerne principalement ce type de contrôle.

L'utilisation conjuguée de ces trois types de mécanismes de détection peut seule permettre d'atteindre, un taux élevé de détection des erreurs. L'efficacité de ces mécanismes implique une structure modulaire du logiciel et un niveau de détection adapté à cette structure : c'est-à-dire que les erreurs doivent pouvoir être détectées et confinées au niveau des modules.

L'intérêt premier d'un tel système de détection est de permettre un accroissement important de la sécurité du logiciel. Par exemple, si le taux de détection est de 0.9, on multiplie par 10 la sécurité du logiciel.

De plus, les mécanismes mis en oeuvre peuvent être utiles avant même l'exploitation du logiciel, lors du codage et des tests. Ils permettent d'accroître la correction du logiciel en contribuant à détecter un plus grand nombre d'erreurs. Ils peuvent également permettre de réduire la durée, donc le coût des tests.

Enfin, par la localisation et le confinement des erreurs au niveau de modules ils facilitent le diagnostic et la maintenance, donc la disponibilité du logiciel.

2.2.2. Recouvrement :

Le recouvrement consiste à neutraliser les conséquences d'une erreur détectée et à assurer le maintien immédiat des fonctions concernées.

La seule technique économiquement raisonnable de recouvrement des erreurs logicielles est celle des "recovery blocks" de Randell (5). Elle repose sur le principe suivant : chaque fonction du logiciel d'application est réalisée par deux modules distincts réalisant chacun la fonction en utilisant un algorithme et/ou des entrées différents de l'autre. Chacun de ces modules est contrôlé par des mécanismes de détection tels que ceux décrits précédemment. Un seul des deux est exécuté en l'absence d'erreurs. En cas de détection pendant son exécution, ce module est abandonné (éventuellement de manière temporaire) et l'exécution de la fonction est tentée avec le second. Une bonne illustration de cette technique pour des fonctions de navigation et de guidage est donnée par H. HECHT (6) ainsi que divers détails pour implémenter les tests de vraisemblance et pour utiliser un watchdog.

Cette technique est évidemment coûteuse : elle double les coûts de programmation et le volume mémoire. En cas de détection d'erreur, la durée d'exécution d'une fonction peut être également doublée.

Elle ne doit donc être utilisée que de manière sélective, uniquement pour des fonctions vitales assurées entièrement par le logiciel.

Cependant, elle ne pose pas de problème particulier de conception (mis à part la nécessité de découvrir deux algorithmes différents pour la même fonction) à partir du moment où le problème de la détection et du confinement est résolu et que l'intégrité des modules de recouvrement est ainsi assurée.

Outre l'accroissement de fiabilité des fonctions redondantes, elle présente des avantages importants sur d'autre plans (7) :

Elle constitue par elle-même une technique de test extrêmement puissante : l'activation séquentielle des deux modules redondants et la comparaison de leurs sorties permet d'automatiser les tests et double leur efficacité. Comme pour la détection, cette technique offre ainsi une retombée intéressante aux plans de la correction du logiciel et des coûts de validation. Elle permet en outre de choisir comme module privilégié à l'exécution celui qui a la durée d'exécution la plus faible, fournissant ainsi un moyen immédiat d'optimisation.

Cette technique, qui ne semble pas être utilisée actuellement dans les applications embarquées, peut se révéler indispensable dans l'avenir si la survie de certaines fonctions aux erreurs logicielles devient nécessaire.

L'analyse des divers moyens qui permettent d'accroître la fiabilité et la sécurité du logiciel (dont on a donné ci-dessus un rapide aperçu), fait apparaître que l'efficacité de ces moyens repose avant tout sur l'existence de mécanismes garantissant l'intégrité du logiciel, c'est-à-dire de mécanismes de contrôle des droits d'accès et d'utilisation (ou plus simplement de contrôle de l'adressage).

C'est de tels mécanismes, développés dans le cadre du projet COPRA, que décrit la suite de cette communication.

3. PROTECTION CONTRE LES ERREURS D'ADRESSAGE

3.1. Critères et contraintes de conception :

- Economie de matériel : Le calculateur COPRA est destiné à des applications embarquées sur avions, engins ou satellites. Le critère d'économie des ressources matérielles reste déterminant, même si les progrès technologiques le rendent moins aigu. En effet, compte tenu de la structure redondante du calculateur - nécessitée pour tolérer les pannes du matériel - tout dispositif matériel, logiciel ou micro-programmé rajouté pour détecter et confiner les erreurs du logiciel, se voit automatiquement appliquer le même degré de redondance, c'est-à-dire est doublé ou quadruplé selon les cas.

- Contraintes de temps : Les applications considérées comportent de fortes contraintes d'exécution en temps réel : durées d'exécution, temps de réponse, etc... La solution retenue doit donc introduire une durée d'exécution supplémentaire minimum. Seuls les contrôles qui ne peuvent être faits à la traduction des programmes doivent être laissés à l'exécution, même si cela conduit à alourdir sensiblement la chaîne de traduction (ce qui n'est pas le cas comme on le montre ci-après).

- Langage de programmation : Les erreurs d'adressage et la protection contre ces erreurs peuvent être différentes selon que le langage de programmation utilisé est de haut niveau ou de type assembleur. Il n'entrait pas dans les objectifs du projet COPRA de définir un nouveau langage évolué. Il faut noter à ce propos que si en théorie un langage évolué semble présenter moins de risques d'erreurs d'adressage qu'un langage assembleur, dans la réalité il existe peu de compilateurs qui effectuent un contrôle complet de l'adressage. De plus, les compilations séparées sont source d'erreurs d'adressage au moment de l'édition de liens.

La solution retenue suppose l'utilisation du langage d'assemblage de COPRA et de l'assembleur et éditeur de liens associés. Elle peut cependant être transposée à un langage évolué existant en ajoutant un précompilateur chargé de traiter les directives de structuration spécifiques du contrôle de l'adressage.

3.2. Structure du logiciel d'application :

L'efficacité de la solution retenue repose essentiellement sur une structure du logiciel bien adaptée aux applications embarquées.

Cette structure résulte d'une double décomposition :

- décomposition en fonctions opérationnelles (appelées "fonction" dans la suite) telle qu'elle est issue habituellement des spécifications fonctionnelles du système et du logiciel. Par exemple, un logiciel d'avion est décomposé en fonctions de localisation, de guidage, de désignation, etc...

- décomposition en processus ou tâches (il n'est pas utile ici de faire la distinction), qui découle des contraintes "temps réel". Le logiciel peut par exemple être décomposé en tâches cycliques devant s'exécuter à fréquence fixe (5 Hz, 50 Hz, etc...), en tâches "immédiates" dont l'activation est commandée par l'occurrence d'événements aléatoires (interruptions) et en tâches différées dont l'activation est gérée par un moniteur logiciel. La décomposition peut être plus ou moins fine, le principe reste toujours le même :

regrouper les parties du traitement ayant des contraintes temporelles d'activation identiques, et établir entre les tâches ainsi définies une hiérarchie d'exécution.

Cette double décomposition porte sur les instructions comme sur les données et conduit à une partition du logiciel en modules, le module étant l'intersection d'une fonction et d'une tâche. Un module appartient à une seule fonction et une seule tâche et est lui-même structuré en procédures.

Une telle décomposition permet de définir trois niveaux principaux de localité des données :

- données locales (ou privées) d'un module, accessibles uniquement par les instructions de ce module.
- données locales d'une fonction, accessibles par les seuls modules de cette fonction.
- données globales, accessibles à l'ensemble des fonctions.

Cette structure est illustrée par la figure 1.

On établit également une distinction entre code et données d'un module ainsi qu'entre constantes et variables.

Avec une telle structure, on dispose de zones de communication bien définies entre les modules d'une même fonction : zone des données locales de la fonction, ainsi qu'entre les fonctions : zones des données globales. Ces zones constituent des interfaces aussi concrètes que celles existant entre fonctions physiquement réparties dans un système distribué.

Des contrôles efficaces basés sur le principe de méfiance mutuelle peuvent être mis en oeuvre à chaque accès à ces interfaces par une fonction ou un module. Ces contrôles peuvent être effectués, par des tests de vraisemblance par exemple, avant qu'une donnée soit écrite dans une zone de communication par une fonction ou un module émetteurs ou après lecture des données dans ces zones par des fonctions ou modules récepteurs.

En d'autres termes, avec une telle structure, les erreurs localisées dans un module ne peuvent être propagées à un autre module de la même fonction ou à une autre fonction que par la transmission de valeurs erronées dans les zones de communication définies ci-dessus. Ainsi, chaque module appartenant à une fonction critique peut être muni de tests d'acceptation des données qu'il reçoit d'une autre fonction (ou d'un module moins critique de la même fonction), de tests de contrôle des données qu'il peut émettre vers le système, et d'un module de recouvrement réalisant le même traitement que lui avec un algorithme et/ou des entrées différentes et qu'il activera lorsque ces tests auront un résultat négatif.

Pour permettre la mise en oeuvre efficace de tels dispositifs (mise en oeuvre qui ne peut être faite qu'au niveau de la conception du logiciel d'application) il faut garantir que les interfaces entre modules et fonctions sont bien respectées. Ceci signifie au minimum qu'aucun module ne doit pouvoir écrire, de manière volontaire ou à la suite d'une erreur de programmation, ailleurs que dans sa zone locale, ou dans la zone des données locales de la fonction à laquelle il appartient, ou dans une zone de données globales. Les lectures en dehors de ces zones ne présentent pas les mêmes dangers, car si elles proviennent d'une erreur de programmation, elles pourront être aisément détectées par les tests sur les sorties du module qui a lu ces données d'entrée erronées. Elles sont par nature non destructives, et leur détection n'est utile que pour renforcer la finesse du niveau de détection.

Un système de détection des écritures dans des zones non autorisées présente l'avantage, outre l'accroissement de fiabilité et de sécurité opérationnelles du logiciel, de faciliter les tests de validation et la maintenance.

3.3. Mécanismes de contrôle

Leur objectif principal est donc d'interdire toute écriture d'un module en dehors des zones autorisées. Il faut noter ici que parmi ces zones non autorisées se trouvent les zones contenant le code des programmes et les données constantes. Celles-ci sont normalement protégées, dans les applications embarquées, par un stockage en mémoire morte. Cependant, si certaines de ces zones se trouvent en mémoire vive, pour une raison ou une autre, les mécanismes de contrôle interdisent de manière aussi efficace toute écriture dans ces zones.

Le contrôle assurant la protection contre les écritures intempestives dans des zones non autorisées est effectué à deux moments distincts :

- lors de l'assemblage des programmes
- lors de l'exécution, soit pendant les tests de vérification et de validation, soit pendant la mission.

Afin de limiter au minimum le matériel, le logiciel et les temps d'exécution supplémentaires nécessaires aux contrôles dynamiques à l'exécution, l'essentiel des contrôles est fait au moment de l'assemblage des programmes.

3.3.1. Méthode de contrôle :

Le mécanisme de base de ces contrôles consiste à vérifier que toute adresse apparaissant dans une instruction d'écriture d'un module est bien une adresse appartenant à l'une des zones de données autorisées à ce module.

Au moment de l'assemblage, seules les adresses non calculées de variables appartenant au même train d'assemblage peuvent être contrôlées.

Les adresses calculées (seul l'adressage indexé existe dans COPRA de ce point de vue) ne sont connues qu'à l'exécution et ne peuvent être contrôlées qu'à ce moment.

Les adresses de variables définies dans d'autres trains d'assemblage ne sont connues qu'à l'édition de liens. Effectuer les contrôles correspondants au moment de l'édition de liens alourdit à la fois l'assembleur et l'éditeur. Plutôt que cette solution, on a préféré la suivante :

Tous les contrôles (hormis ceux des adresses calculées) sont réalisés par l'assembleur au moment de l'intégration des différents modules et fonctions. Ceci consiste à réunir la totalité du programme en un seul train d'assemblage et à activer les contrôles de l'assembleur à ce moment par une simple carte OPTION qui commande l'exécution des passes supplémentaires nécessaires au contrôle.

Ceci présente l'avantage de pouvoir effectuer les nombreux assemblages et éditions lors des étapes de codage et de tests partiels avec un assembleur et un éditeur normaux. Le seul coût supplémentaire n'est introduit que lorsque sont faits les quelques assemblages (très peu sont nécessaires) du programme complet, et ce coût est limité en raison de la facilité avec laquelle peuvent être faits les contrôles, l'assembleur disposant alors de toutes les informations nécessaires.

3.3.2. Contrôles réalisés à l'assemblage

L'élément fondamental qui permet de réaliser les contrôles définis précédemment est que tant au niveau du programme écrit en langage d'assemblage que des mécanismes d'adressage de la machine virtuelle, la structure définie précédemment (fonctions, tâches, modules, données globales) apparaît de manière explicite.

Le langage d'assemblage contient des directives permettant de déclarer chacun des objets précédents et d'associer chaque module à une seule fonction et une seule tâche.

Ainsi, toute donnée déclarée est associée explicitement :

- soit à une zone de données globales, auquel cas toutes les instructions peuvent adresser cette donnée.

- soit à une zone de données locales à une fonction (les déclarations de ces données suivent la déclaration de la fonction). Dans ce cas l'assembleur vérifie (lorsque l'option contrôle est active) qu'aucune instruction située dans un module n'appartenant pas à la fonction n'adresse cette donnée. Les modules appartenant à la fonction sont déclarés à la suite de la déclaration de la fonction et sont donc connus de l'assembleur.

- soit à une zone de données locales à un module. Ces données sont déclarées dans les diverses procédures composant le module. L'assembleur vérifie que les seules instructions d'écriture adressant ces données appartiennent aux procédures du module.

Si cela s'avère utile, les contrôles précédents pourront facilement être étendus aux instructions de lecture.

D'autre part, le langage d'assemblage permet de faire la distinction entre procédures internes d'un module et procédures "bibliothèque" pouvant être appelées à partir de modules différents. Ces dernières ne peuvent adresser directement que leurs données locales, affectées dynamiquement à l'exécution, et qui appartiennent alors aux zones de données locales des modules appelants. Ces procédures de bibliothèque peuvent évidemment travailler sur d'autres zones de données, mais uniquement lorsque les adresses de celles-ci leurs sont passées sous forme de paramètres 'adresse' lors de leur appel.

Les déclarations de ces procédures et leurs instructions d'appel sont vérifiées par l'assembleur pour contrôler que les paramètres effectifs ne peuvent pas donner de droit supplémentaires aux modules appelants. De plus, l'assembleur vérifie que les procédures internes d'un module ne peuvent être appelées à partir d'autres modules.

Des restrictions plus classiques, comme la limitation de la portée des étiquettes définies dans une procédure à cette seule procédure sont également vérifiées par l'assembleur.

Enfin, celui-ci génère automatiquement les modes d'adressage en fonction de l'emplacement des données adressées ainsi que les longueurs des différentes zones de données. Ces deux points permettent la mise en oeuvre des contrôles à l'exécution.

3.3.3. Contrôles réalisés à l'exécution :

Ces contrôles concernent les seules adresses calculées à l'exécution. Le seul mode d'adressage calculé dans COPRA est l'adressage indexé. L'adressage indirect est réservé à l'adressage des paramètres passés à une procédure, mais les adresses effectives de ces paramètres sont contrôlées à l'assemblage en vérifiant que les adresses de tous les paramètres déclarés dans les appels de procédure d'un module donné sont bien autorisées à ce module.

L'adressage indexé est possible à la fois pour les instructions traitant les données et pour les instructions de branchement.

- Adressage indexé des données :

Les seuls modes d'adressage autorisés aux programmes d'applications sont l'adressage basé direct et l'adressage basé direct indexé. Le calculateur dispose de trois registres de base spécialisés :

Registre L : Ce registre sert à l'adressage des données locales d'un module. Il est géré automatiquement lors des instructions d'appel et de retour des procédures de ce module, et le programme n'y a pas accès. Le contenu de L pointe donc toujours sur le 1er mot de la zone des données locales du module courant. L'assembleur génère automatiquement une adresse basée par L pour toutes les instructions référant les données locales du module.

Registre G : Il sert à l'adressage des données locales d'une fonction. De la même manière que pour L, Il est géré automatiquement à chaque fois qu'une tâche appelle un module appartenant à une nouvelle fonction. Il pointe donc toujours le 1er mot de la zone des données locales de la fonction courante. L'assembleur génère également une adresse basée par G pour toutes les instructions référant les données locales de la fonction.

Registre Y : permet d'adresser les données globales. Ce registre peut être chargé par une instruction spéciale disponible aux programmes d'application. Ceci permet de pointer différentes zones de données globales. L'assembleur vérifie cependant que les instructions de chargement de ce registre ne réfèrent que des zones de données globales et génère une adresse basée par Y pour toutes les instructions référant des données globales. Ce registre pointe donc toujours le premier mot de la zone de données globales en cours d'utilisation.

On retrouve ainsi, au niveau de la machine virtuelle, des dispositifs de structuration reflétant exactement la structure du logiciel décrite plus haut.

L'assembleur génère automatiquement, à l'adresse précédant immédiatement le 1er mot de toute zone de données, la longueur de cette zone.

Le contrôle des adresses indexées de données consiste simplement à vérifier que l'adresse calculée est comprise entre le contenu du registre de base utilisé, et ce même contenu augmenté de la longueur de la zone. Pour des raisons d'économie en temps d'exécution, il n'est effectué que lors des instructions d'écriture. Il est donc très peu coûteux (1 cycle mémoire supplémentaire pour les instructions d'écriture indexées, et 1 mot supplémentaire par zone de donnée). Il ne demande aucun matériel spécifique dans les unités de traitement. Son efficacité tient à la cohérence existant entre la structure du logiciel et les outils de la machine virtuelle.

- Branchements indexés :

Les adresses de branchement indexé réfèrent des tables de branchements déclarées comme telles dans la procédure. L'assembleur vérifie que toute instruction de branchement indexé fait référence à une telle table. Il génère automatiquement la longueur d'une table dans le mot précédant celle-ci.

À l'exécution, la microprogrammation spécifique de ce mode d'adressage vérifie que le contenu du registre d'index est positif et inférieur à la longueur de la table.

Ce mécanisme est également très peu coûteux et évite toute déviation par rapport au graphe de contrôle tel qu'il a été vérifié à l'assemblage. Il est indispensable pour garantir l'efficacité de tous les autres contrôles, car en effet, tout branchement erroné pourrait conduire à exécuter des actions dangereuses non soumises aux contrôles prévus.

4. CONCLUSION

La solution dont les caractéristiques générales ont été présentées dans cette communication est en cours de développement. Son efficacité doit faire l'objet d'une validation comprenant des essais sur des programmes dans lesquels auront été volontairement introduites des erreurs. Ces essais sont prévus.

Il faut également s'assurer que la structure rigoureuse qu'elle implique pour les programmes est suffisamment "naturelle" pour ne pas être ressentie comme une contrainte au niveau de la conception et de la programmation.

Elle constitue la base d'un système de tolérance aux erreurs du logiciel, et elle est la condition préalable à la mise en oeuvre de mécanismes de détection spécifiques de chaque logiciel d'application, ainsi que des fonctions de recouvrement, également spécifiques.

REFERENCES :

1. C. MERAUD, F. BROWAEYS "COPRA - Une ligne nouvelle de calculateurs reconfigurables ultrafiabiles". Paper n° 62 of this symposium
2. P.J. DENNING "Fault-tolerant operating Systems
ACM Computing Surveys, Vol. 8, N° 4, Dec. 1976, p 359
3. D.M. ENGLAND "Capability concept mechanisms and structure in system 250".
Proc. Colloque IRIA Août 1974 "Protection in Operating Systems" p 63.
4. L. BOI, P. MICHEL Design and Principles of a fault-tolerant system.
Proc. of 3rd International Conf. on Software Engineering ATLANTA 1978
p 207.
5. B. RANDELL "System Structure for Software Fault Tolerance".
Proc. International Conference on Reliable Software, 1975.

A REDUNDANT INERTIAL NAVIGATION SYSTEM FOR IUS

by

R. A. Baum - Hamilton Standard Division
of United Technologies
1690 New Britain Avenue
Farmington, Connecticut 06032 - U.S.A.

G. E. S. Morrison - Boeing Aerospace Company
P.O. Box 3999
Seattle, Washington 98124 - U.S.A.

R. C. Peters - The Aerospace Corporation
P.O. Box 92957
Los Angeles, California 90009 - U.S.A.

SUMMARY

This paper describes a high-performance strapdown Redundant Inertial Navigation System (RINS) being developed for the Space Transportation System Inertial Upper Stage (IUS) and presents an overview of the test results obtained to date. The IUS RINS is the first navigation system specifically designed to provide a fully redundant configuration with self-contained redundancy management algorithms which will permit the system to experience an in-flight failure and then automatically detect, isolate, and eliminate the failed element and continue the mission without loss or out-of-specification degradation of the guidance function.

INTRODUCTION

The IUS is being developed by the Boeing Aerospace Company under contract to the USAF Space and Missile Systems Organization (SAMSO) for use as an upper stage on the National Aeronautics and Space Administration (NASA) Space Shuttle and USAF Titan 34D vehicles. Technical support is provided to the IUS program by the Aerospace Corporation. The Hamilton Standard Division of United Technologies is developing the IUS Redundant Inertial Navigation System (RINS) in response to USAF/Boeing specifications. The RINS consists of a Hamilton Standard redundant inertial measurement unit and Delco Electronics computers.

The IUS program entered Full Scale Development (FSD) in January 1978. The FSD program includes the design, qualification, and production of the first eleven flight systems. Component qualification is scheduled for completion in 1979. First flight aboard a Shuttle vehicle is scheduled for early 1981.

Hardware Design

The IUS RINS consists of an internally redundant, skewed, computationally cross-strapped, five sensor (designated A-B-C-D-E) Redundant Inertial Measurement Unit (RIMU) and dual redundant Computer Units (CU) arranged in the redundant configuration shown in Figure 1. The RINS system design satisfies IUS reliability level and single point failure tolerancy requirements in the most cost effective manner. The ability to realize the full reliability potential of the RINS is based on the development of effective redundancy management techniques, both individually within the RIMU and Computer Units, and at the RINS system level.

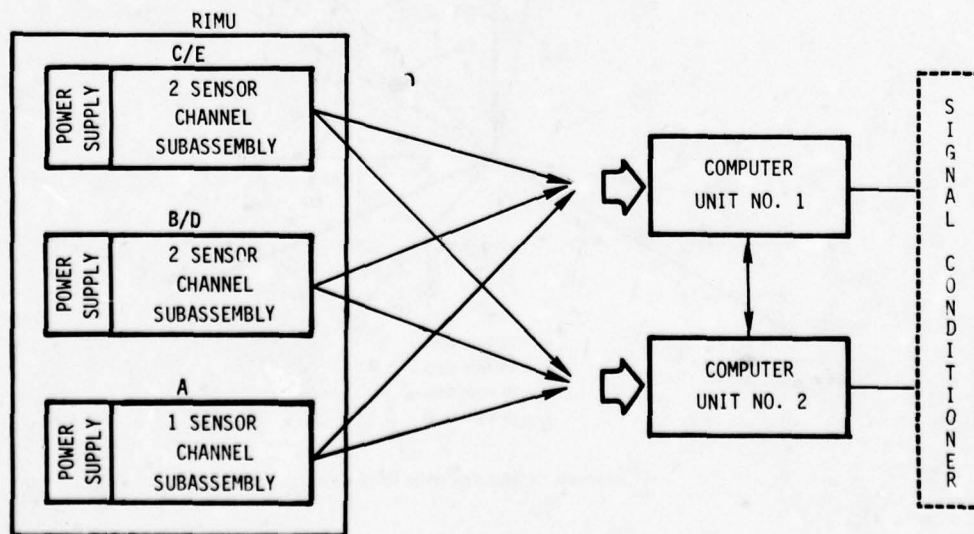


FIGURE 1 - IUS RINS REDUNDANT CONFIGURATION

RIMU

The RIMU design utilizes five (5) Hamilton Standard RI-1010 single-degree-of-freedom, rate integrating, gas bearing, gyros and five (5) Kearfott 2401 single-axis, linear accelerometers arranged in a "skewed-conical" array on a single cluster assembly. The basic conical geometry is depicted in Figure 2. The vector directions of the sensor input axes can be arranged, after translation, symmetrically about the surface of a cone whose half angle is arc cosine $1/\sqrt{3}$. In the RIMU, the cone axis is tilted from the X-RIMU axis toward the Z-RIMU axis by a rotation of approximately 41.2° about the Y-RIMU axis (see Figure 3). This orientation places two gyro output axes parallel to the thrust (X-RIMU) axis. In addition, all gyro spin axes and accelerometer pendulous axes are perpendicular to the thrust axis. A compact placement of the sensors in the cluster assembly is achieved by means of linear translations of the input axis vector directions within the skewed conical geometry.

RIMU SENSOR INPUT AXES

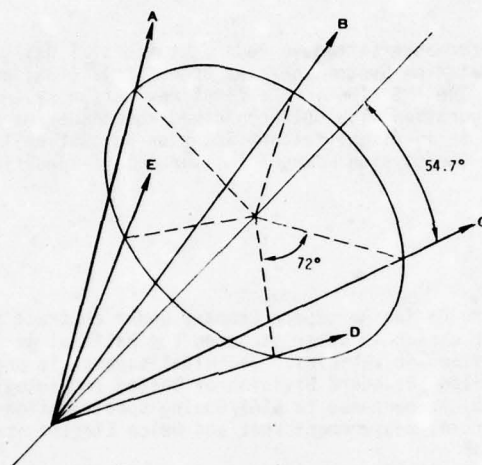
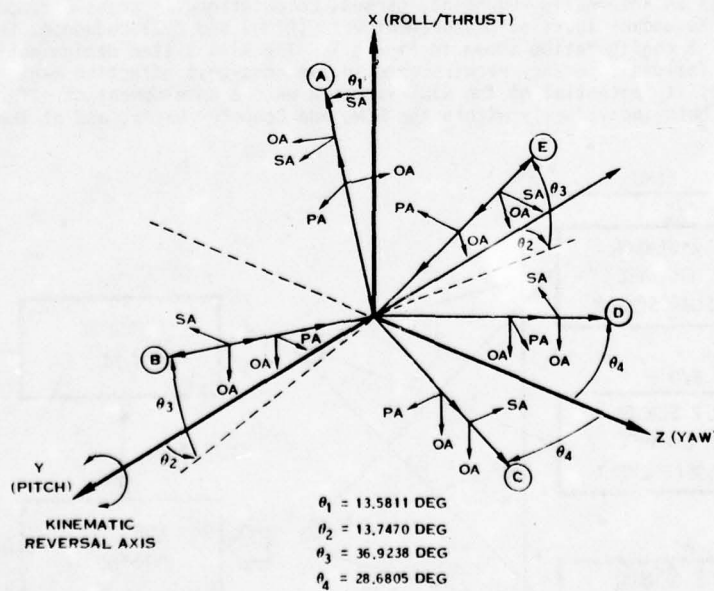


FIGURE 2



SKEWED CONICAL SENSOR AXES ORIENTATIONS

FIGURE 3

The sensor assembly is supported by three fully autonomous identical sets of electronics which provide conditioned power, digital control, thermal control, synchronization and the necessary computer interfaces for each set of inertial sensors which are partitioned in a 2:2:1 configuration. This design implementation ensures single failure tolerance by providing three fully independent sensor channels. A block diagram of a RIMU channel is shown in Figure 4.

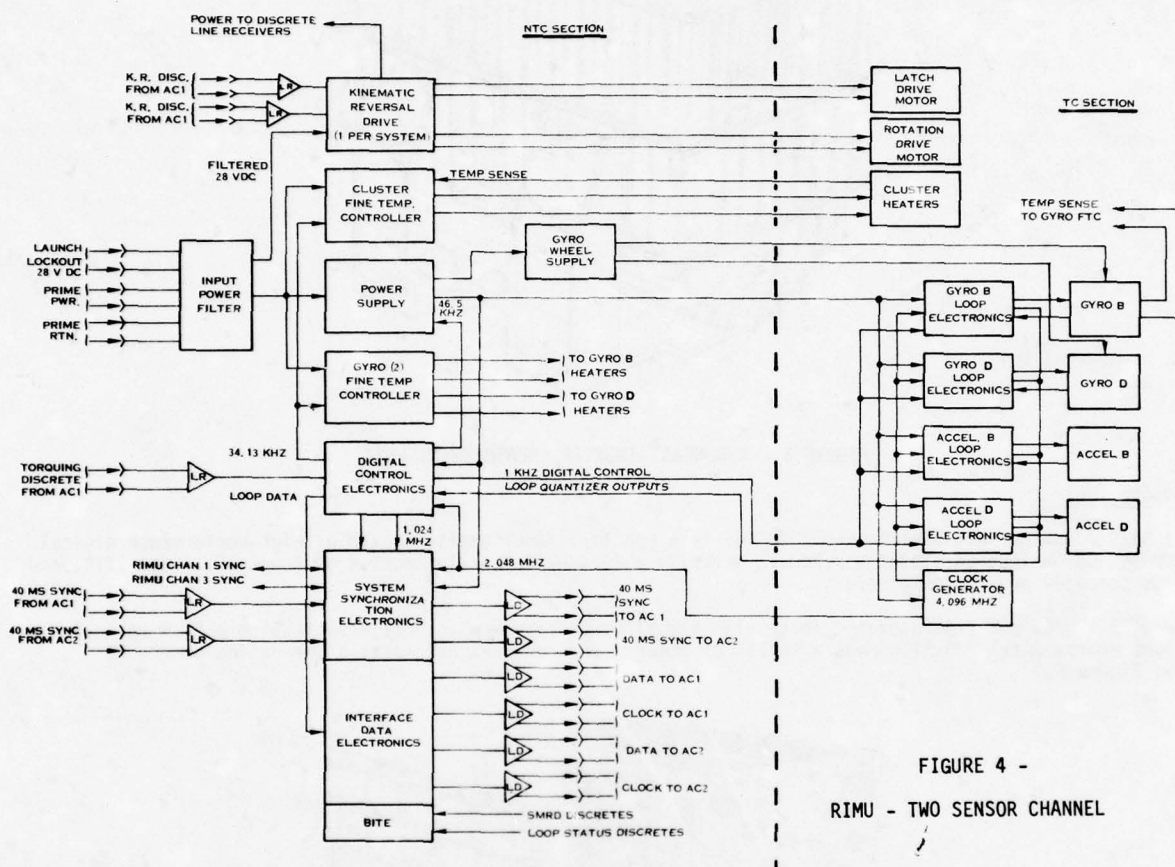


FIGURE 4 -
RIMU - TWO SENSOR CHANNEL

The gyro loops are scaled for a +30 degree per second range, and the accelerometer loops are scaled for +20 g's. This operational range is achieved using a pulse width modulated, binary, rebalance mechanism which produces a digital output proportional to incremental angle (0.42 arc sec/pulse) and velocity (0.0025 fps/pulse) for the gyro and accelerometer loops, respectively. The gyro and accelerometer rebalance electronics are designed to provide matched loop responses with a bandwidth of 84 Hz.

Data from each of the five RIMU gyro and accelerometer sensors is simultaneously sent to both Computer Units under control of a collectively generated system synchronization pulse. Seventeen data words are transmitted each 10 milliseconds from each of the three RIMU channels. Included in each transmission are raw sensor pulse count data, sensor temperature data, critical electronics temperature data and power supply levels, all in digital format. Analog levels representing temperature and voltage levels are converted to ten bit digital format in each of the three RIMU channels. This design implementation allows transmission of all inertial data, telemetry data and BITE status through a single 1.024 MHz data link for each RIMU channel, thereby reducing interface complexity.

A "Kinematic Reversal" mechanism provides the capability of rotating the sensor cluster about a horizontal axis through an automatic sequence of four orientations, spaced 90 degrees apart. This is used during earth prelaunch operation to perform a precise, self-contained alignment and calibration of the RIMU in order to achieve the best possible mission accuracy. The sensor cluster is tightly secured at its nominal 0° position prior to launch.

The RIMU measures 43.0 cm x 28.0 cm x 40.0 cm, including fins, weighs 36.3 kg and consumes 96 watts of steady state sensor and electronics operating power. The unit is designed to maintain precise active internal temperature control over an external temperature range of -20°C to +56°C. A maximum 100 watts of environment dependent supplementary heater power is required to maintain precise temperature control of the inertial sensors and temperature sensitive elements of the RIMU electronics throughout the specified IUS mission operating temperature range. The RIMU external configuration is shown in Figure 5.

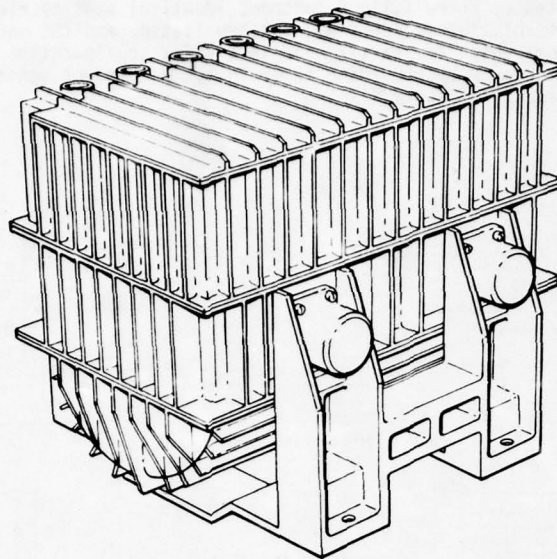


FIGURE 5 - REDUNDANT INERTIAL MEASUREMENT UNIT

Computer Unit

The 362S Computer Unit used in the IUS RINS is a modular, functionally flexible, high-performance digital computer which incorporates the hardware maturity gained by Delco Electronics on the F-16, Titan IIIC, and Delta computer production programs.

The 362S IUS/Computer is packaged in an aluminum alloy enclosure which measures 36.5 cm x 36.5 cm x 19.3 cm, weighs approximately 24 kilograms with 65K of memory, and consumes 222 watts of operating power. (See Figure 6.)

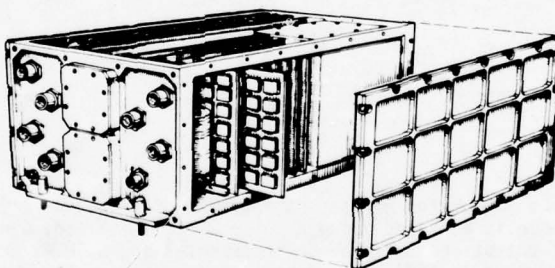


FIGURE 6 - COMPUTER UNIT

Primary design features include:

- o High throughput (600 KOPS plus)
- o 65K word x 16 bit low power CMOS memory
- o Fixed/floating point processor
- o Extensive support software library including JOVIAL compiler
- o Growth potential (memory, instructions and I/O)

The M362S is a microprogrammed, high-speed, general-purpose, parallel computer employing 16- and 32-bit instructions and 16-, 32- and 64-bit data words. Thirty-two bit memory words are used for extended instructions and floating point and double precision data words. Arithmetic operations are binary, with negative numbers in the two's complement form. The processor is mechanized with standard medium scale integration (MSI) and large scale integration (LSI) transistor-transistor logic (TTL) integrated circuits.

The M362S Stretched Processor is an expansion of the existing processor assembly used in the Delco F-16 Fire Control M362F computer. The mechanical, thermal and packaging designs are identical to those successfully utilized by Delco on the USAF Titan IIIC Universal Space Guidance System and Delta Inertial Guidance System programs. The bus techniques used for internal communications allow both memory expansion and the addition of special input/output modules without impacting the processor, memory circuit configuration or the power supply.

Redundancy Management

RIMU

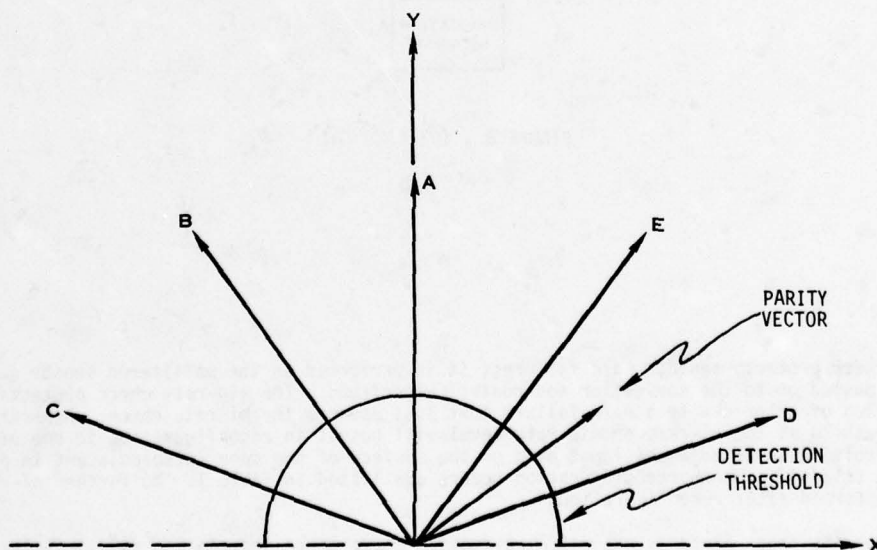
The selected approach to the redundancy management/failure detection aspects of the RIMU design satisfies the IUS reliability level and RINS single failure tolerant criteria while providing growth potential for additional levels of failure detection and fault isolation. The choice of a computationally cross-strappped five sensor array provides the ability to detect at least one sensor channel or one power supply failure. Combined hardware built in test equipment (BITE) and computer software-implemented diagnostics are used to detect RIMU subassembly failures. The key to successful RIMU operation is the ability to detect and isolate sensor performance degradation (i.e., "soft") failures. Previous experimental efforts have shown that theoretically obvious soft failures are, in practice, difficult to detect. The problem is that, if treated only in real time, the magnitude of sensor errors which would ultimately degrade injection accuracy after propagating throughout a mission are small and tend to be masked by noise. The solution to this problem is a unique adaptation of the Mid-Value Selection principle which employs a series of three parallel tests performed at different computational frequencies to process longer term filtered data, as well as real time data, to significantly increase the probability of detection and reduce false alarm rate.

The FDI algorithm utilizes a series of threshold checks upon the magnitude of "parity vectors" formed from the gyro and accelerometer data. The known relative placement of the inertial sensors permits the writing of linear dependence equations for the sensor outputs. These linear dependence equations define the equivalence, or parity, among the sensor outputs and, as such a failure in any sensor will cause a deviation from the expected parity.

For the pentad array of sensors, the associated parity space is two dimensional, and the parity algorithm is relatively simple. An illustration of the derived "parity vector" is presented in Figure 7. The amplitude of the parity vector is a measure of the inconsistency in the redundant sensor data. The parity vector is directed toward the placement of the failed sensor input axis in parity space.

Failure detection and pair isolation is signaled when the parity vector amplitude exceeds a minimum value beyond which a failure is considered to have occurred. The strapdown computations are then executed using only data from the failure-free triad (ABC in the example of Figure 7). Failure isolation is signaled when the parity vector amplitude is sufficiently large and points in sufficient proximity to a particular sensor axis. Subsequent strapdown computations are then executed using data from the remaining quartet of sensors. When no failures are indicated, the strapdown computations gain the performance advantage provided by combining data from the full pentad sensor array.

RIMU FDI VIA MID-VALUE SELECTION IN PARITY SPACE



PARITY VECTOR = MEASURE OF REDUNDANT DATA INCONSISTANCY

FIGURE 7

The parity vector checks are performed at three different computational rates: high-rate (50/sec), mid-rate (25/sec), and low-rate (2/sec), are illustrated in the block diagram of Figure 8. Adaptive thresholds are employed at each of the levels of parity checks. The thresholds are defined in terms of a constant plus a function of the mean square angular rate and linear acceleration environments. Parity vector filtering is performed at 50/sec for both the mid-rate and low-rate checks; the filtering is much heavier on the latter than the former. The threshold forcing functions for these two checks receive the same filtering as their parity vectors.

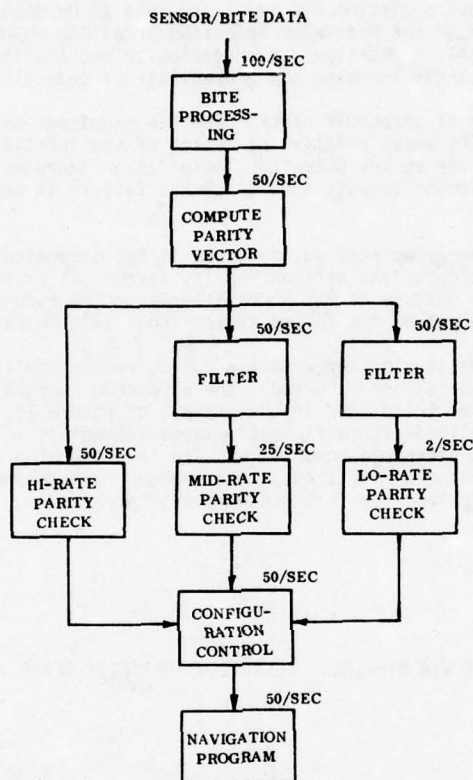


FIGURE 8 - IUS RIMU FDI

The hi-rate check protects against hard failures; it is performed on the unfiltered sensor data before the data are passed on to the navigation and control algorithms. The mid-rate check protects against the accumulation of error due to a hard failure that just escapes the hi-rate check. A parity vector which exceeds a threshold at the hi-rate or mid-rate level will result in reconfiguration to one of five triads of sensors involving three adjacent input axes on the surface of the cone (also adjacent in parity space). The "adjacent triads" to which reconfiguration occurs are listed in Table I. No further hi-rate or mid-rate checks are performed after reconfiguration.

The low-rate parity check protects against a navigation performance degradation (soft) failure. A parity vector which exceeds the low-rate threshold will first result in reconfiguration to one of the five adjacent triads. The low-rate parity checks continue, however, and if an isolation decision function is satisfied, reconfiguration to the remaining quartet occurs (see Table I). The sensor isolation decision function is a combination of thresholds on both the amplitude and phase angle of the parity vector, coupled with an N-count (the number of steps in low FDI cycle).

After a parity vector check has caused reconfiguration to a quartet, no further parity checks are made unless the software is reset by an external command.

BITE processing monitors the critical voltages, temperatures, loop closure discretes, gyro spin motor rotation detector (SMRD) discretes, and quality of data transmission from all three RIMU channels. There are a total of 15 voltages (5 per channel), 9 temperatures (5 gyro, 2 cluster and 2 case), 10 loop closure discretes, 5 gyro spin motor rotation detection (SMRD) discretes and 33 data quality bits (11 per channel) to be processed. Fresh values of the voltages and discretes are transmitted to the computers at a 100/sec rate. The temperatures are transmitted at a lower rate. The BITE processing results in sensor BITE flags which are made available at the 50/sec rate to assist in the hi-rate sensor configuration selection. An N-count is used to prevent spurious noise from initiating a permanent reconfiguration due to BITE. The BITE flag is coded such that reconfiguration to any one of the 15 redundant subsets can be quickly accomplished.

A RIMU channel failure is detected by means of the hi-rate voltage checks. The sensor assignments per channel are as follows: A-gyro and A-accelerometer in channel 1, B and D-gyros and B and D-accelerometers in channel 2, and C and E-gyros and C and E-accelerometers in channel 3. The assignments are such that a channel failure will result in reconfiguration to a quartet or one of two "non-adjacent triads", as shown in Table II. This provides the best performance in the degraded mode of operation.

TABLE I
RIMU RECONFIGURATION AFTER SENSOR LOOP FAILURE

<u>Failed Sensor Loop</u>	<u>Redundant Subsets</u>	
A	BCD CDE	BCDE
B	CDE ADE	ACDE
C	ADE ABE	ABDE
D	ABE ABC	ABCE
E	ABC BCD	ABCD

TABLE II
RIMU RECONFIGURATION AFTER CHANNEL FAILURE

<u>Failed Channel</u>	<u>Redundant Subsets</u>
1	BCDE
2	ACE
3	ABD

The fault isolation technique has been implemented and demonstrated during the current RIMU test program. The mechanization affords the following advantages over others considered:

- o Maintenance of the ability to achieve the maximum performance advantage through the processing of redundant data when operating in the failure free mode.
- o Equal sensitivity of error detection for each instrument channel.
- o Self-healing (i.e., automatic reconfiguration) without loss of data for transient errors.
- o Minimum software memory storage and execution time requirements.

Computer Unit

Built-in test equipment (BITE) in the computers and a computer-resident self-test program provides monitoring points for failure detection and isolation. In the event of a failure, the fault is identified and the failed computer is automatically switched out of the system. The navigation process continues in the redundant computer and failure mode data from the failed computer is available for telemetering. The failure modes associated with redundancy management of the IUS computers are listed below with the corresponding interrupts generated in the computer:

<u>FAILURE MODE</u>	<u>COMPUTER INTERRUPT</u>
Memory Error	Hardware
Illegal Memory Store	Hardware and Software
System Synchronization Failure	Hardware
Computer I/O Failures	Hardware and Software
GMT Input Circuit Failure	Software
Interrupt Processing Failure	Software
Measurement Data Failure	Hardware and Software
Control Status Failure	Software

Accomplishments to date

All design-critical and/or potential technical risk elements of the RINS design are being evaluated in an effort to provide the substantiated technical baseline needed to promptly initiate and efficiently execute the final design effort in accordance with the demanding schedule established for the IUS program. Major RINS related activities have included:

- o Optimization of the RIMU sensor orientation, self-contained alignment and calibration procedures, and the FDI algorithms
- o USAF/CIGTF evaluation of Hamilton Standard RI-1010 gyros and Singer Kearfott Model 2401 accelerometers with digital loops
- o Fabrication of engineering prototype RIMU's
- o Fabrication of engineering prototype M362S Delco Computers
- o Checkout of RIMU self-calibration/alignment, navigation and FDI algorithms on M362S Computer
- o Laboratory demonstration of M362S throughput capability

Analyses and software programming activities are virtually complete. The prototype RINS has undergone laboratory tests, van navigation tests, and FDI laboratory demonstrations. Significant test results obtained to date include:

CIGTF Tests

RI 1010 gyro and 2401 accelerometer evaluation tests were conducted at the Central Inertial Guidance Test Facility (CIGTF), Holloman Air Force Base, New Mexico. These tests, performed under USAF/SAMSO-Aerospace Corporation direction, were designed to provide an independent, detailed, critical evaluation of candidate inertial instruments during the Boeing Aerospace Company competitive IUS/INS Validation Phase program.

The CIGTF test sequence included the following tests:

Stability Performance - Continuous run and across shutdowns

Random Vibration - 15.7 g rms, 3 minutes per axis, equivalent vehicle input environment

Shock Spectrum - 170 g's peak @ 150 Hz, 100 g's to 104 Hz

A summary of the demonstrated performance capability of the inertial sensors across the complete CIGTF test program including all environments, shutdowns, remountings, etc., is given below:

o Gyro Stability Across Thermal, Shock, and Vibration

Scale Factor	27 - 40 ppm
Bias	.002 - .01 °/hr
MUSA	.012 - .03 °/hr/g
MUIA	.004 - .016 °/hr/g

o Accelerometer Stability Across Thermal, Shock, and Vibration

Scale Factor	35 - 73 ppm
Bias	8 - 35 µg

Computer Throughput - RINS Computer Unit throughput was demonstrated at the Delco Electronics, Santa Barbara Operations, Goleta, California, facility using a solid state memory M362S Stretched Processor Computer. A prime contractor defined instruction mix representative of boost vehicle guidance and control computational requirements was used for this demonstration. The throughput achieved was 702 KOPS, exclusive of hardware implemented memory Error Detection and Correction (EDC) logic. Conservative analytical estimates of the effect of the addition of EDC indicate an operational capability in excess of 600 KOPS.

Laboratory Navigation Tests - Diversified laboratory evaluation tests were conducted in a program specifically designed to demonstrate critical functional, redundancy management and software elements of the RIMU design as well as the integrated system level inertial performance capability of the unit. The test sequence included a formal series of alignment and static and dynamic navigation tests selected to extract an accurate assessment of the performance capability of the RIMU in operational usage. The test program was successful in demonstrating:

- o Ability of factory calibration procedures to extract mission critical inertial sensor calibration coefficients for the skewed conical pentad sensor array.
- o Self-calibration (i.e., Kinematic Reversal) and alignment algorithms used in conjunction with the selected redundant sensor array.
- o Baseline inertial sensor/navigation performance incorporating enhancements realized through use of redundant algorithms and the selected redundant sensor cluster array geometry.
- o Inertial sensor/navigation performance capability in reduced sensor array configuration.
- o Fault Detection and Isolation (FDI) algorithms.

Table III summarizes the results of Hamilton Standard's RINS Laboratory Test Program. The data presented include separate alignment and combined alignment/navigation runs selected to excite potential major system error sources. The alignment tests included static and simulated vehicle sway motion environments. The navigation test series included repeated static, 90 degree rotation, Scorsby motion, X and Y vehicle axis oscillation, low level linear and angular vibration, and extended duration (30 hour) turn/dump tests. These test data significantly bettered IUS mission error budget values and established high confidence in the ability of the RIMU to satisfy IUS/RINS operational requirements.

TABLE III
NAVIGATION PERFORMANCE SUMMARY

<u>NAVIGATION</u>		
<u>DURATION</u>	<u>TEST</u>	<u>MEAN CEP RATE (NM/HR)</u>
3 HR	STATIC NAV.	0.323
3 HR	90° NAV	0.922
12 HR	SCORSBY, 3°, .1 HZ	0.843
12 HR	SCORSBY, 2°, .1 HZ	2.081
30 HR	TURN AND DUMP NAV	0.915
85 MIN	OSCILLATION NAV	1.094
85 MIN	OSCILLATION NAV, SCORSBY	0.765
85 MIN	LINEAR VIBRATION	0.615
85 MIN	ANGULAR VIBRATION	2.516
12 HR	360° PITCH ABOUT Y AXIS	0.901
12 HR	360° ROLL ABOUT X AXIS	1.412
12 HR	360° ROTATION ABOUT Z AXIS	0.562
1 HR	POINTING/SLEW TESTS	0.675
<u>ALIGNMENT</u>		
<u>TEST</u>		<u>HEADING DEVIATION ARC SEC (3σ)</u>
STATIC ALIGNMENTS		168.02
SWAY ALIGNMENTS		179.70
RSS WEIGHTED AVERAGE		169.05

Mobile Van Navigation Tests - Following completion of the laboratory test program, the RINS was installed in Hamilton Standard's Inertial System Test Van for a brief series of evaluation tests conducted in the dynamic environment generated by the linear and angular motion of the test van. Three-hour duration tests were conducted over a 7.5 mi. course laid out on local secondary roads. Each test was initiated with a 60-minute self-contained static alignment which was followed by a continuous navigation period during which the vehicle was alternately driven and parked. The selected sequence was chosen to emulate boost vehicle operation. Three tests were conducted as summarized below. The mean value of the CEP rate of the three test series is 0.78 nm/hr.

o RINS Van Test Summary

Run #1	1.2 nm/hr
Run #2	0.65 nm/hr with FDI operational
Run #3	0.48 nm/hr with FDI operational

FDI Demonstration - The RIMU FDI demonstration consisted of a series of RINS dynamic navigation tests conducted on a Goertz 3-axis automatic test stand. During these tests, the system was implemented with a full complement of RIMU FDI algorithms and was exercised both without and in the presence of a series of computer-introduced failure scenarios. The test program was successful in demonstrating the detection and isolation of the following failures:

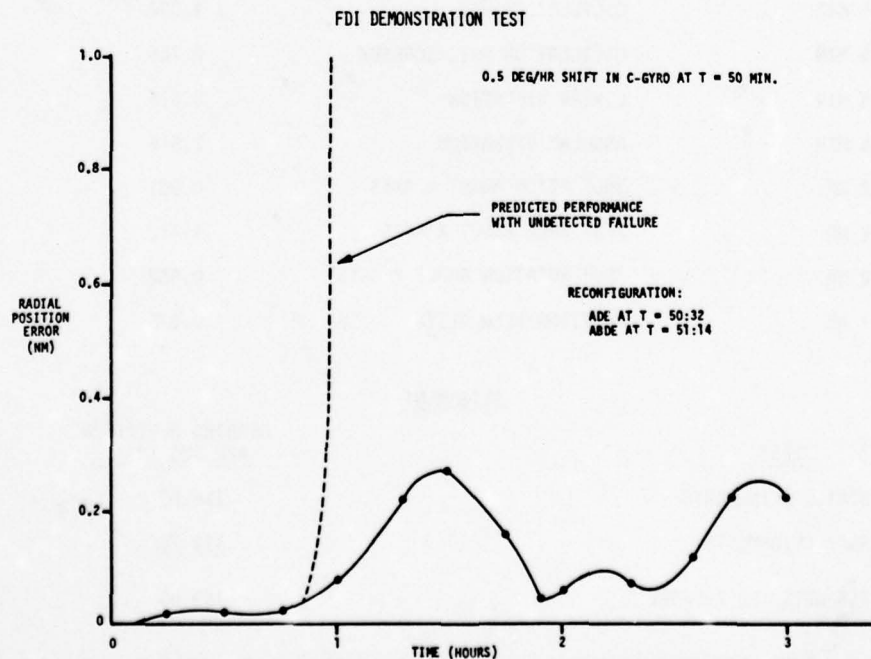
- o Sensor failures during quiescent operation (both soft and hard)
- o Power supply failures during quiescent operation
- o Sensor failures during linear vibration (both soft and hard failures)
- o Sensor failures during angular oscillations (both soft and hard failures)
- o Power supply failures during linear vibration

In all of the above instances, following the occurrence of a purposely introduced failure, the system automatically reconfigured to the proper sensor triad or quartet configuration, as appropriate.

Initial FDI demonstration navigation runs were made without pre-programmed failures in order to evaluate selected detection threshold levels in terms of false alarm occurrences. Each navigation test was initialized by means of a 20-minute static self-alignment. The first runs consisted of a series of slews and dumps about each body-axis at 5 deg/sec, with time spent in quiescent operation after each slew and dump. This permitted the RINS to remain in off-nominal orientations with respect to gravity, thus exercising all gyro mass unbalance and accelerometer scale factor parameters. No false alarms were observed during this test sequence.

The runs which followed incorporated +5 deg/sec oscillations at 6 Hz about the RINS X-axis, initially with no failures programmed in order to provide a reference, and then with a series of programmed failures. No false alarms were observed in the first case, and correct isolation was observed in the latter cases.

The final FDI demonstration test consisted of a quiescent (static) navigation run with a simulated C-gyro channel failure of 0.5 deg/hr programmed to occur at the 50-minute mark. The system correctly isolated the channel subassembly pair and then the sensor, with only a slight degradation in terrestrial navigation performance as shown in Figure 9.



CONCLUSIONS

Developmental tests conducted with IUS RINS hardware in 1977 and 1978 have demonstrated the ability to provide a fully redundant strapdown navigation system which can detect a failure, isolate and eliminate the failed element, and continue to perform without the loss or out-of-specification degradation of the guidance function. Operational deployment of the IUS will mark the first time that such a navigation system has been used for space vehicle applications.

REFERENCES

1. S290-22118, Boeing/SAMSO Prime Item Development Specification for Inertial Measurement Unit, 9 March 1978.
2. S290-22119, Boeing/SAMSO Prime Item Development Specification for Computer Subsystem, 9 March 1978.
3. D290-10021-1, Boeing Specification for IUS Inertial Measurement Unit and Avionics Computer - Validation Phase, 1 April 1977.
4. HSER 6909, Hamilton Standard IUS CIGTF Gyro Test Unit, 30 September 1977.
5. HSER 6917, Hamilton Standard IUS NTA Laboratory Test Program, 21 November 1977.
6. HSER 6919, Hamilton Standard IUS NTA FDI Demonstration Test Program, 6 February 1978.
7. ADTC-TR-78-29, Volume II, IUS Component Tests - Hamilton Standard Model RI-1010 Gyro and Singer Kearfott Model 2401 Accelerometer - Central Inertial Guidance Test Facility, 6585th Test Group; Holloman Air Force Base, New Mexico, May 1978.

DEFINITION OF THE HIERARCHICAL NETWORK FOR AGGRESSIVE ENVIRONMENTS (RHEA)

M.BUIS J.C.LAPRIE J.MARCO D.R.POWELL

Laboratoire d'Automatique et d'Analyse des Systèmes
du Centre National de la Recherche Scientifique,
7 Avenue du Colonel Roche,
31400 TOULOUSE, FRANCE.

ABSTRACT

This paper describes the transmission level of the dependable data communication support system RHEA intended for the reliable and survivable interconnection of data processing units (subscribers) in an aggressive environment.

The functional and dependability specifications of this system lead to the definition of a hierarchical architecture using a network-structured, damage and fault-tolerant global transmission medium and star-structured, fault-tolerant local transmission media. The contention method is used for control of access to the transmission media.

A standard transmission-medium/subscriber interface is defined that implements an associative addressing scheme in order to cater for data broadcasting modes and facilitated dynamic system reconfiguration.

Throughout the design, a top-down approach has been adopted with the use of qualitative and quantitative evaluations at each level in order to make motivated choices among the different possible solutions.

INTRODUCTION

The aim of the hierarchical network for aggressive environments (RHEA: "Réseau Hiérarchisé pour Environnements Agressifs") is to provide a reliable and survivable communication support system for the interconnection of loosely-coupled subscribers (computing elements, sensors, actuators) so as to form a dependable distributed avionics processing system.

The dependability objectives of a distributed processing system must be satisfied at three levels (figure 1):

- the user level (resource management): measurement of the state of the resources in function of the system objectives,
- the transport level: realization of a secure packet protocol and management unit,
- the signal level: realization of a reliable and survivable transmission medium and its associated access control scheme.

A previous study [1] led to the definition of a two-level transport level structure as shown in figure 2. The lower level of this structure consists of several local transmission media used to interconnect affinity group clusters of physically localized subscribers. The upper level of this structure is made up of a global transmission medium used to interconnect the clusters of the lower level and the physically distributed subscribers. The two levels of transmission media may be studied independently as each level is seen as a subscriber from the other level.

The constraints that are imposed on the design of the transmission media are as follows:

- (a) They must tolerate single component failures, electrical perturbations and, where applicable, multiple local failures due to physical damage.
- (b) All functions must be decentralized so as to eliminate any possible hard-core.
- (c) One-to-all communication must be possible with a single transmission in order to allow synchronous data sampling and facilitated dynamic reconfiguration.

This paper is divided into three parts:

- the first part deals with the control of access to the transmission media,
- the second part concerns the definition of the physical structures of the transmission media,
- the third and last part is dedicated to the definition of the interface between the transmission media and the subscribers.

I. CONTROL OF ACCESS TO THE TRANSMISSION MEDIA.

This section of our paper describes a study that we have carried out in order to choose a method of controlling access to the transmission media of RHEA.

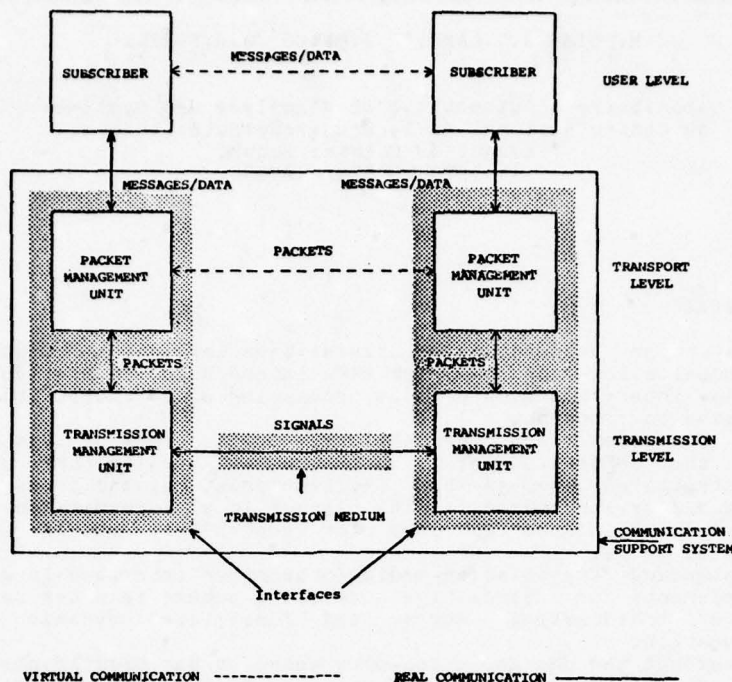


Figure 1: Inter-subscriber communication path

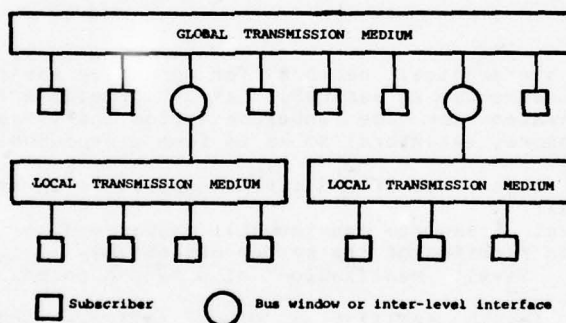


Figure 2: The hierarchical transport level structure

I.1. Totally decentralized control methods.

In accordance with the design goal of total decentralization of all functions, we have studied techniques for decentralized control of access to the transmission media with a view to a comparison of their functional and operational performances.

The two most likely candidates for decentralized access control are:

- decentralized daisy chain control [3][4],
- contention control [5][6].

I.1.1. Decentralized daisy chain control.

The decentralized daisy chain access control technique is an asynchronous time division multiplexing method in which a transmission frame is defined by allocating to each subscriber one or more positions in a cyclic transmission frame.

From an implementation viewpoint, each subscriber possesses a read-only register in which each bit at logic '1' represents a frame position allocated to that subscriber. A cross-section of all these read-only registers shows each frame position allocated to only one subscriber. Each subscriber possesses a counter that addresses that subscriber's register. This counter is incremented at the end of each transmission on the transmission medium and indicates to the subscriber the instant at which it may transmit. If the subscriber does not wish to transmit when it is its turn then it sends a synchronization packet that enables the other subscribers to increment their counters.

Figure 3 gives the algorithm of this access control method and figure 4 represents the timing relationship between four subscribers sharing a common transmission medium using this technique.

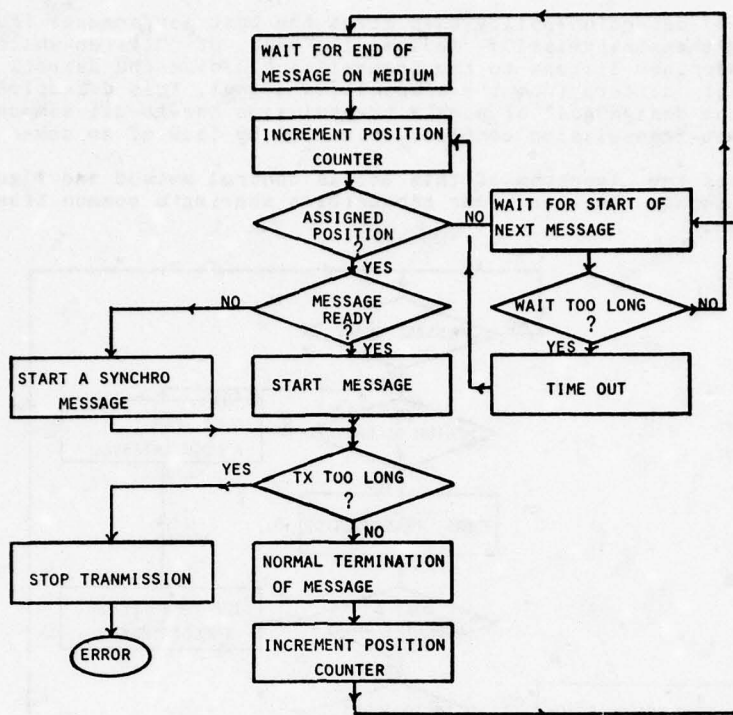
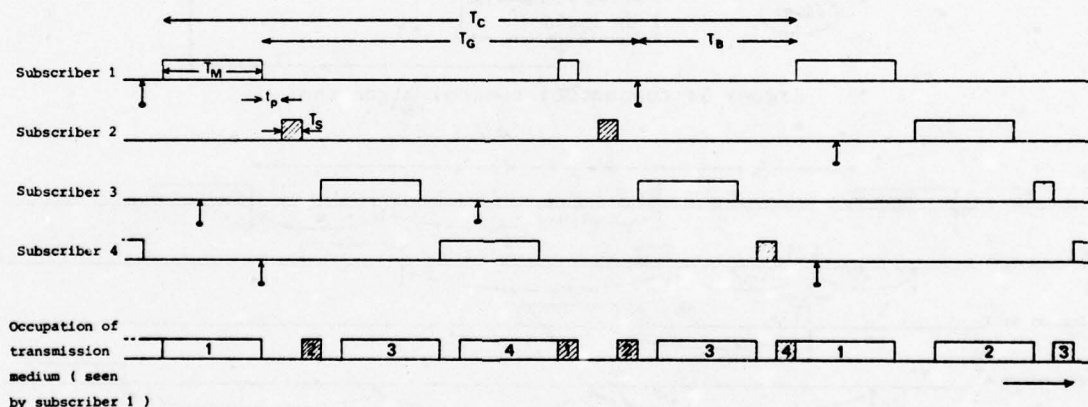


Figure 3: The decentralized daisy chain algorithm

ASSUMPTIONS

- . 4 subscribers with one place each in frame.
- . Identical propagation time between each pair of subscribers (star structure).
- . Fixed length packets.

NOMENCLATURE

- T_B : Blocking time
 T_C : Cycling time
 T_G : Time to new packet generation
 T_M : Packet length
 T_S : Synchronization packet length
 t_p : Propagation time

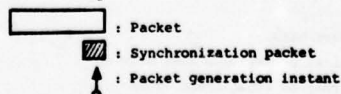


Figure 4: Timing relationship between four subscribers controlled via a decentralized daisy chain

I.1.2. Contention control.

The principle of contention control resides essentially on the following three points:

- as soon as as subscriber has a packet ready then it may attempt to transmit,
- a subscriber may not transmit if it detects that the transmission medium is occupied; this is the only restriction concerning access rights,
- conflicts due to transmissions starting within one propagation time must be detected and lead to retransmissions of the contending packets after random delays.

The contention detection policy that gives the best performance from the viewpoint of occupation of the transmission medium is that of "listen-while-talk". Whilst transmitting, a subscriber listens to the transmission medium and detects a conflict when the received signal differs from the transmitted signal. This detection policy is also compatible with the design goal of single transmission one-to-all communication (on the contrary to a post-transmission conflict detection by lack of an acknowledgment from a specified receiver).

Figure 5 gives the algorithm of this access control method and figure 6 represents the timing relationship between four subscribers sharing a common transmission medium using this technique.

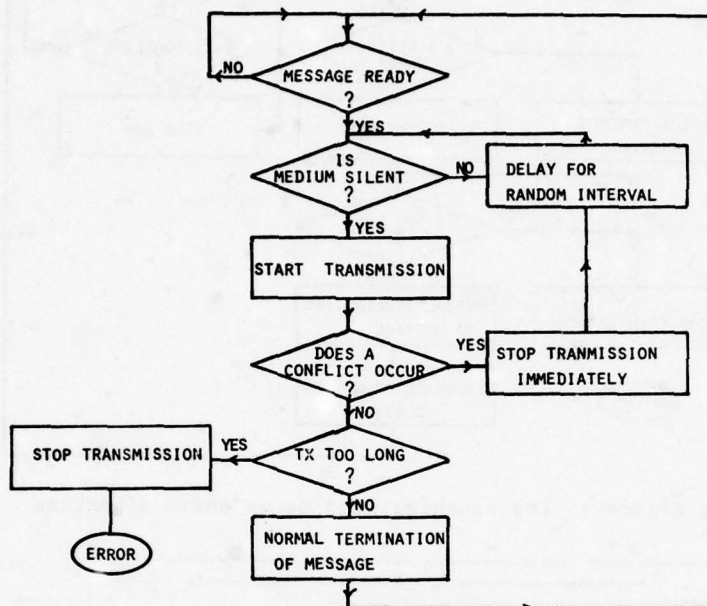
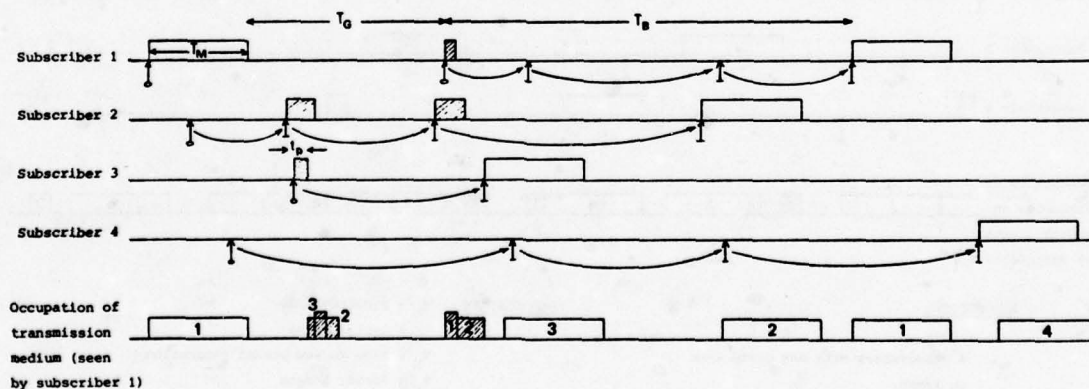


Figure 5: Contention control algorithm



ASSUMPTIONS

- 4 subscribers.
- Identical propagation time between each pair of subscribers (star structure).
- Fixed length packets.
- Instantaneous detection of conflicts.

NOMENCLATURE

- T_B : Blocking time
 T_G : Time to new packet generation
 T_M : Packet length
 t_P : Propagation time

- : Correctly transmitted packet
 : Packet interrupted by a conflict
 : Packet generation instant
 : Transmission re-try instant
 : Random delay

Figure 6: Timing relationship between four subscribers using contention control

1.2. Functional performance evaluation.

In order to evaluate the functional performance of the two envisaged decentralized access control methods, we have carried out discrete-time simulations using the IBM GPSS package [7].

So as to be able to objectively compare the performances of each method, a certain number of common assumptions were adopted:

- thirty-two subscribers are interconnected via a single transmission medium,
- the subscribers generate fixed length packets,
- a subscriber is said to be "blocked" (in that it may not generate further messages) from the moment it generates a message until the correct transmission of the corresponding packet,
- when a subscriber becomes unblocked, the time until the generation of a new message is exponentially distributed.

It was further assumed that the propagation times between each pair of subscribers are identical. This supposition simplifies the simulation and will facilitate future comparison with analytical models of the access control methods. Although this means that the transmission medium is supposed to have a star structure, the validity of the evaluation is upheld for all structures for which the propagation time is small compared to the length of the transmissions.

1.2.1. Decentralized daisy chain control.

The main results of the decentralized daisy chain control simulation are illustrated by figure 7 which shows the mean blocking time (δ) and the mean cycle time (γ) (normalized with respect to the packet length) in function of the "normalized generation frequency" (λ) defined as the ratio of packet-length to the mean generation time.

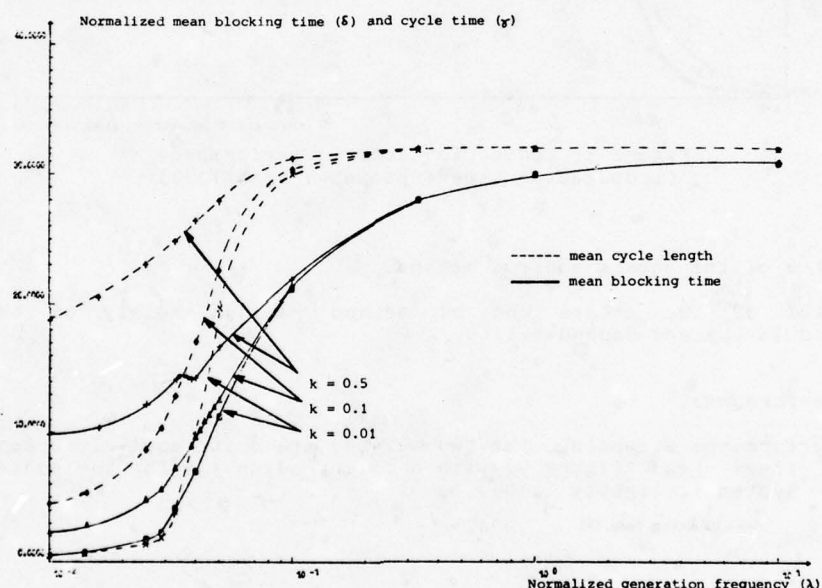


Figure 7: Decentralized daisy chain performance
(propagation time = packet-length/1000)

The curves are drawn for three values of 'k' that represents the ratio of the lengths of the synchronization packets and ordinary packets.

Several interesting phenomena may be observed:

- δ increases suddenly for $\lambda \approx 1/32$; this value of λ corresponds to the onset of transmission medium saturation,
- for a lightly loaded system (small λ), δ greatly depends on the synchronization packet length,
- for $k=0.01$, the mean cycle length can be less than the mean blocking time; this apparently contradictory phenomenon can be explained by the fact that there is a correlation between successive cycles: a cycle containing a packet is more likely to be followed by another cycle containing a packet than one containing only synchronization packets,
- for $k=0.5$, the mean blocking time is a non-monotonic function of λ ; this is due to a change of operating regime where the mean cycle time is approximately equal to the mean generation time.

1.2.2. Contention control.

The normalized mean blocking time (δ) for the contention control method is shown in figure 3 in function of the normalized generation frequency for different values the mean retransmission frequency (σ) defined as the ratio of packet-length to mean retransmission period.

We may observe from these curves that:

- δ presents a similar behavior as for the decentralized daisy chain in that the onset of saturation occurs for $\lambda \approx 1/32$,

- reaches an asymptotic value of 31 for high generation frequencies,
 -for the considered value of the propagation time (packet-length/1000), has only a small influence on $\bar{\epsilon}$.

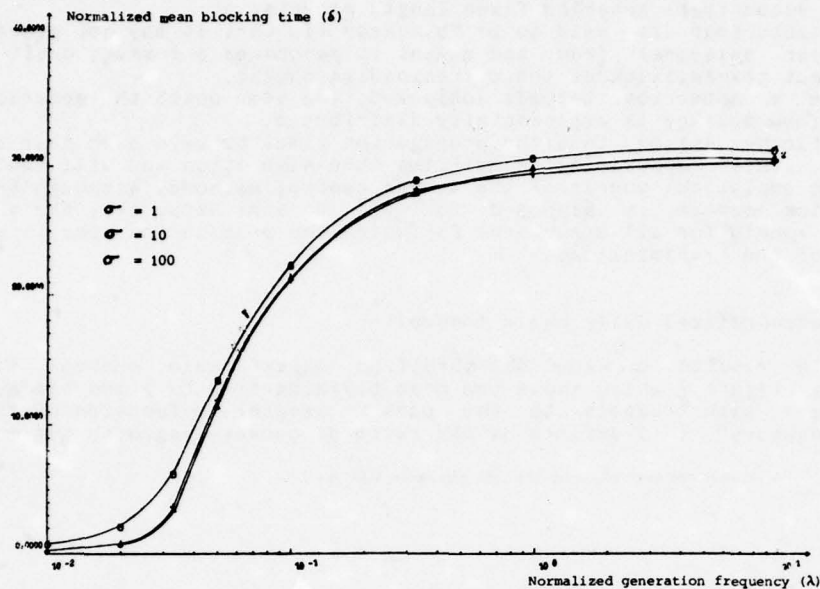


Figure 3: Contention control performance
 (propagation time = packet-length/1000)

I.3. Choice of the access control method.

The choice of the access control method resides mainly on the criteria of performance, modularity and dependability.

I.3.1. Performance.

From a performance viewpoint, the two methods are quite equivalent concerning their mean blocking times (see figure 9) with a slight advantage for the contention control method when the system is lightly loaded (small λ).

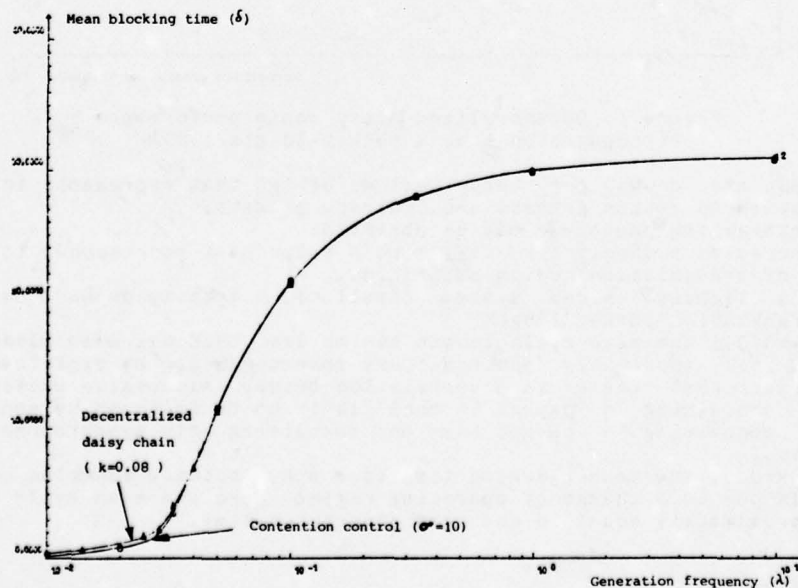


Figure 9: Performance comparison of the two access control methods

It could be argued that this comparison of mean blocking times is not sufficient and that whereas the daisy chain method has a bounded blocking time (the maximum cycle time), the contention method has a theoretically unbounded blocking time. However, it is our opinion that if one specifies a maximum blocking time of say 100 times the mean blocking time then the probability of this maximum time being exceeded is negligible compared to that of system failure due to other physical faults.

1.3.2. Modularity.

From a modularity viewpoint, the contention control method has a clear advantage over the daisy chain method in that all the subscriber interfaces are identical and do not require any programming as to the number of subscribers in a transmission frame. This aspect has very important consequences on system extendability and maintainability.

1.3.3. Dependability.

From a dependability viewpoint, a qualitative analysis disfavors the daisy chain control method since in this method conflicts can occur if ever the counters in each interface become desynchronized. Thus, special circuitry must be included to detect conflicts and to resynchronize the system. Since conflicts are supposedly rare then these circuits will be mainly idle and any latent faults could remain undetected.

The contention control method, on the contrary, displays a regular behavior where conflicts are a natural event. All of the corresponding circuitry is thus used regularly and may thus be easily tested.

1.3.4. Conclusion.

We terminate this study of decentralized transmission medium access control methods with the conclusion that the contention control method displays the best properties and this method has thus been chosen for both levels of transmission media in the RHEA system.

II. DEFINITION OF THE TRANSMISSION MEDIA.

As it was stated in the introduction to this paper, the two levels of transmission media may be studied independently, i.e. they may be treated as global "bus" structures [8] for which the physical implementation must be carefully chosen in order to obtain the required dependability.

The upper, global transmission medium is intended for the interconnection of geographically distributed subscribers. Thus physical damage due to a hostile environment must be taken into account, leading possibly to multiple localized failures. The only structure permitting multiple localized failures without total disruption of communication is a multipath or network structure. Due to the physical constraints imposed by the geographical location of the subscribers, the topology of the network is usually irregular.

The lower, local transmission media are intended for the interconnection of geographically localized subscribers. In this case, the fact as to whether or not the environment be hostile is irrelevant since any external perturbation could affect all of the system as easily as a part of it. The star structure does not present in this case any wiring problem and it seems to be the best structure if the central node can be designed in such a way that no single failure leads to total loss of communication.

An example topology of the RHEA structure showing both levels of transmission media (irregular network and star) is given in figure 10.

II.1. The global transmission medium: a network structure.

II.1.1. Classification of decentralized multipath structures.

The choice of a network structure for the physical implementation of the global transmission medium is due to the multipath properties of such a structure. In the context of this study, and in accordance with the design goals of (a) total decentralization of all functions and (b) the provision of the possibility of one-to-all transmissions, we are neither interested in networks containing any centralized control [9] nor in message or packet-switching networks that do not possess a broadcast facility.

Figure 11 gives a classification of the solutions that have been studied in the context of this study. All of the solutions are compatible with fiber-optic inter-node links which enables both high E.M.I. immunity and galvanic isolation of the subscribers.

The first level of this classification concerns the type of network i.e. passive or active. It should be noted that in the case of an active network the nodes must be powered and the corresponding electrical supply system should have survivability features that are homogeneous with those of the transmission medium [7].

The second level of the classification concerns the entity which must be routed (in active networks):

- pulses,
- signals (pulse trains whose coded information is irrelevant to network operation),
- packets (pulse trains whose coded information is essential to network operation).

The third level of classification concerns the technique that is used to enable transmitting subscribers to detect conflicts.

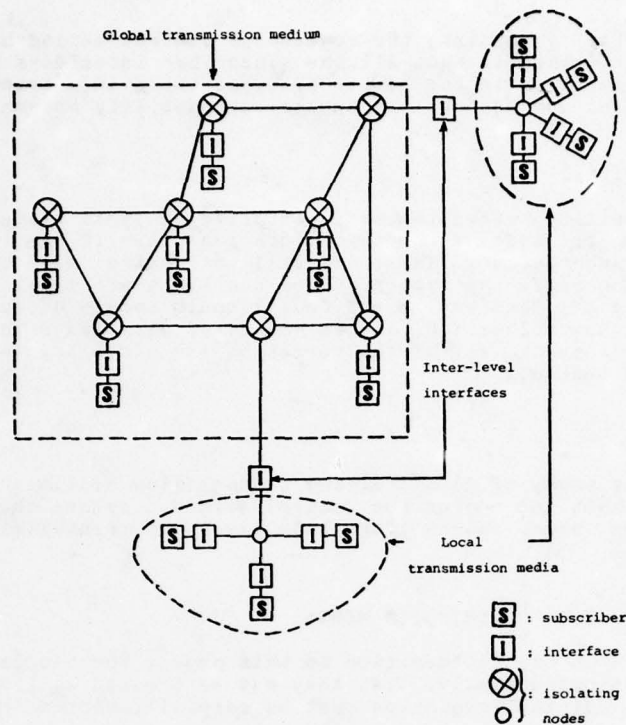


Figure 10: An example RHEA topology

II.1.1.1 Passive optical nodes.

The first and simplest multipath transmission medium uses passive n-way optical couplers as nodes. Such a solution has two important restrictions:

- there is a severe compromise between link length and baud-rate due to the multiplicity of optical paths,
- the number of nodes in the network rapidly introduces prohibitive attenuation and too wide a dynamic range at the optical receivers.

II.1.1.2. Pulse regenerating nodes.

This active network solution resides on the use of a specific coding technique (monopolar return-to-zero) in order to regenerate pulses at each node thus avoiding pulse spreading. Such a technique can be envisaged if the inter-node round-trip propagation time is less than one pulse width. A baud-rate/link-length compromise is thus necessary.

In this particular solution, conflicts are propagated to the contending subscribers by ensuring that each pulse has completely crossed the network before another is launched. There is thus a compromise between baud-rate and total network propagation time. This leads to restrictions as to the number of nodes and the total diameter of the network.

Long packets can be transmitted efficiently through such a network using a double baud-rate technique:

- a slowly-transmitted header to resolve contentions,
- a fast data portion containing the informative part of the packet.

II.1.1.3. Pulse regenerating nodes with conflict detection.

This solution uses nodes that are similar to the previous solution but the nodes now have the responsibility of detecting conflicts.

The compromise between baud-rate and network diameter is thus removed at the expense of increased node complexity (the nodes must detect discrepancies on their inputs and saturate their outputs for a short period so that neighboring nodes also detect a discrepancy) and a lack of node transparency as seen by the subscribers (the subscribers no longer see a logical 'OR' of the transmitted pulses).

There remains the compromise between baud-rate and link length that is inherent to all the pulse routing techniques.

II.1.1.4. Pulse arbitrating nodes with conflict detection.

The use of a multiplexer together with an arbiter circuit enables pulse routing using any sort of signal coding. At the beginning of each set of pulses received on the

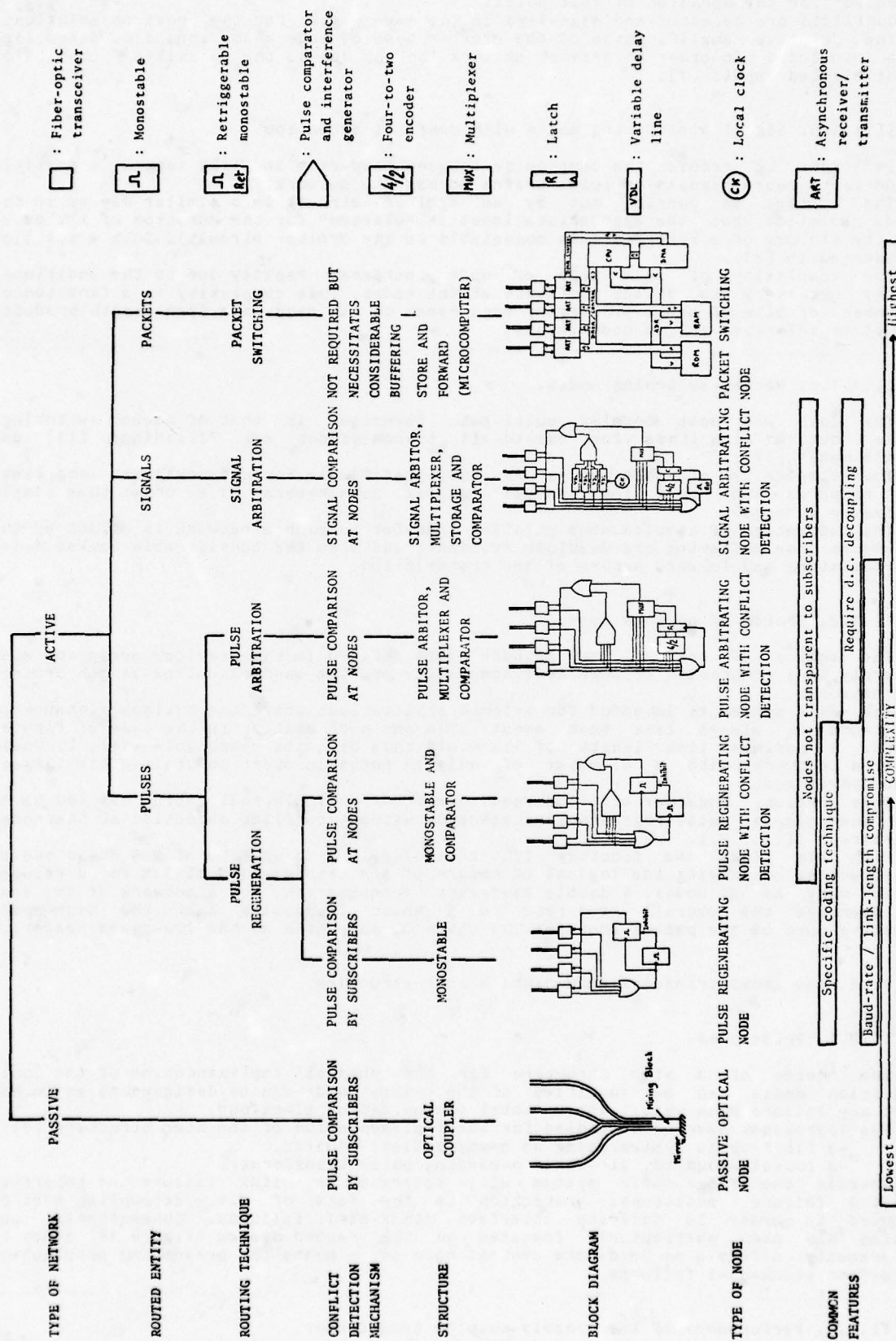


Figure 11: Classification of decentralized multiple path structures

node inputs, a decision is taken as to which input received a pulse first and this input is "elected" for the duration of that pulse.

Conflicts are detected and signalled in the same way as for the previous solution.

The positive amplification of the arbitor type of node means that d.c. decoupling must be included in order to prevent network lock-up due to the parasitic formation of loops of elected inputs [7].

II.1.1.5. Signal arbitrating nodes with conflict detection.

In order to remove the compromise between baud-rate and link length, a possible solution is to route signals or pulse trains across the network.

The routing is carried out by an arbitor circuit in a similar way as in the previous solution but the appropriate input is "elected" for the duration of the pulse train (by the use of a retriggerable monostable in the arbitor circuit). Such a solution was presented in [2].

The complexity of this type of node increases rapidly due to the additional circuitry necessary to detect conflicts at the nodes. This complexity is a function of the number of bits of phase difference (and hence on the baud-rate link-length product) that must be tolerated at the node inputs.

II.1.1.6. Packet-switching nodes.

The last and most complex multi-path technique is that of packet switching. Suitable routing algorithms for one-to-all transmissions are "flooding" [10] and "Huygen's mode" [11].

The complexity of such nodes is only justifiable for networks with long links and/or networks that only interconnect complex subscribers (i.e. other than simple actuators or sensors).

The advantage of simultaneous parallel transfer in such a network is offset by the requirements for buffering and deadlock avoidance and also the considerable packet delay due to the store and forward nature of the transmission.

II.1.2. Choice of network type.

The choice of one of the network types defined in the previous paragraph must necessarily be a compromise between node complexity and the baud-rate link-length product of the network.

The RHEA system is intended for avionic applications where the maximum distance in the network is always less than about 100m and much smaller in the case of fighter aircraft. A maximum link length of 10m would thus be quite reasonable since it would require a network with a diameter of only 10 nodes in order to achieve the largest distance envisaged.

This reason, together with the requirement for a simple realization has led us to choose the simple pulse regeneration technique without conflict detection at the nodes (cf paragraph II.1.1.2.).

With 10m links and Schottky TTL technology, a baud-rate of 0.5 Mbaud can be achieved whilst conserving the logical OR nature of the transmission medium for a network with as many as 25 nodes. A double baud-rate technique can, for a network of the same size, increase the overall baud-rate to 5 Mbaud (supposing that the high-speed informative part of the packet contains 100 times as many bits as the low-speed header).

II.2. The local transmission medium: a star structure.

II.2.1. Principles.

The choice of a star structure for the physical implementation of the local transmission media can be justified if the central node can be designed so as to not exhibit any failure mode that leads to total communication black-out.

Two approaches have been studied for the implementation of the star structure [2]:

- a fiber-optic system using an n-way optical coupler,
- a loosely-coupled, air-cored n-winding pulse transformer.

Whereas the fiber-optic system will tolerate any link failure or interface stuck-at-0 failure, additional protection in the form of d.c. decoupling must be introduced in order to tolerate interface stuck-at-1 failures. Consequently, our attention has been particularly focussed on the second system (figure 12) since it simultaneously offers a no hard-core central node and a means for preventing propagation of interface stuck-at-1 failures.

II.2.2. Performance of the loosely-coupled transformer.

A detailed analysis of the equivalent circuit of a 3-winding loosely-coupled transformer (figure 13) enables us to obtain the transfer function between a pair of windings that is independent of the state (normal or short) of the other winding, i.e., with a coupling coefficient $k \ll 1$, $R_2' = R'$ and $R_3' = 0$ or R' , we obtain:

$$\tilde{V}_2 / \tilde{V}_1 = (pk/LC) (p+1/CR) / (p^2 + p/CR + 1/LC) / (p^2 + p/CR + 1/LC)$$

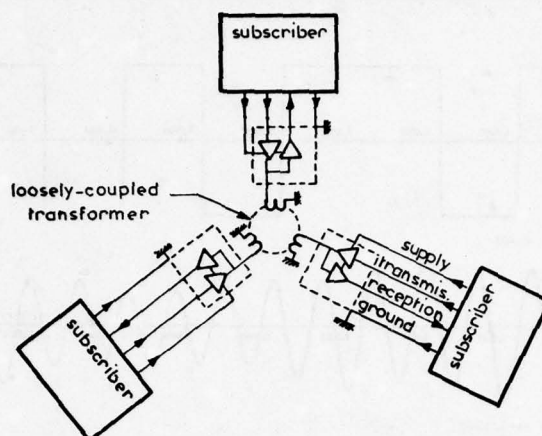


Figure 12: The star structure (configuration shown for 3 subscribers)

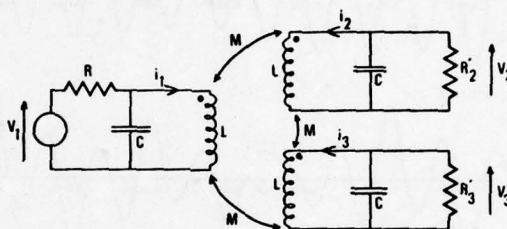


Figure 13: Transformer equivalent circuit

The loose coupling does of course lead to a considerable attenuation and distortion of the transmitted signals but as shown in curve 5 of figure 14, a workable signal can be obtained.

III. DEFINITION OF THE SUBSCRIBER-TRANSMISSION MEDIUM INTERFACE

The aim of the interface between each subscriber and its corresponding transmission medium is twofold:

- management of the actual transmission (transmission level),
- management of the packets (transport level).

The dependability of RHEA imposes certain interface characteristics:

- the provision of means to carry out dynamic system reconfiguration,
- a high reliability of each interface with respect to the total system.

In order to be able to carry out dynamic system reconfiguration, it is advantageous to use an associative addressing mode that enables an abstraction of the actual physical locations of the different calculation, sensing and actuation processes.

The reliability of the interface and of the transmission is imposed in the sense that no failure at this level must have any effect on the correct continuous operation of the overall system.

The logistic constraints of low manufacturing and maintenance costs have led us to define a standard interface that is specialized by program at system initialization time. This programming must specifically take into account the nature of the corresponding subscriber:

- "intelligent" subscribers (e.g. microcomputers) which can carry out calculation functions and/or a partial or overall control of the system,
- "dumb" subscribers (e.g. certain sensors, actuators and visualizations).

The functional structure of this standardized interface is shown in figure 15.

This section of our paper is divided into two sub-sections dealing respectively with:

- transmission management,
- packet management.

III.1. Transmission management

The transmission management part of the interface must carry out all those functions related to the signals vehiculed by the transmission medium:

- encoding and decoding,

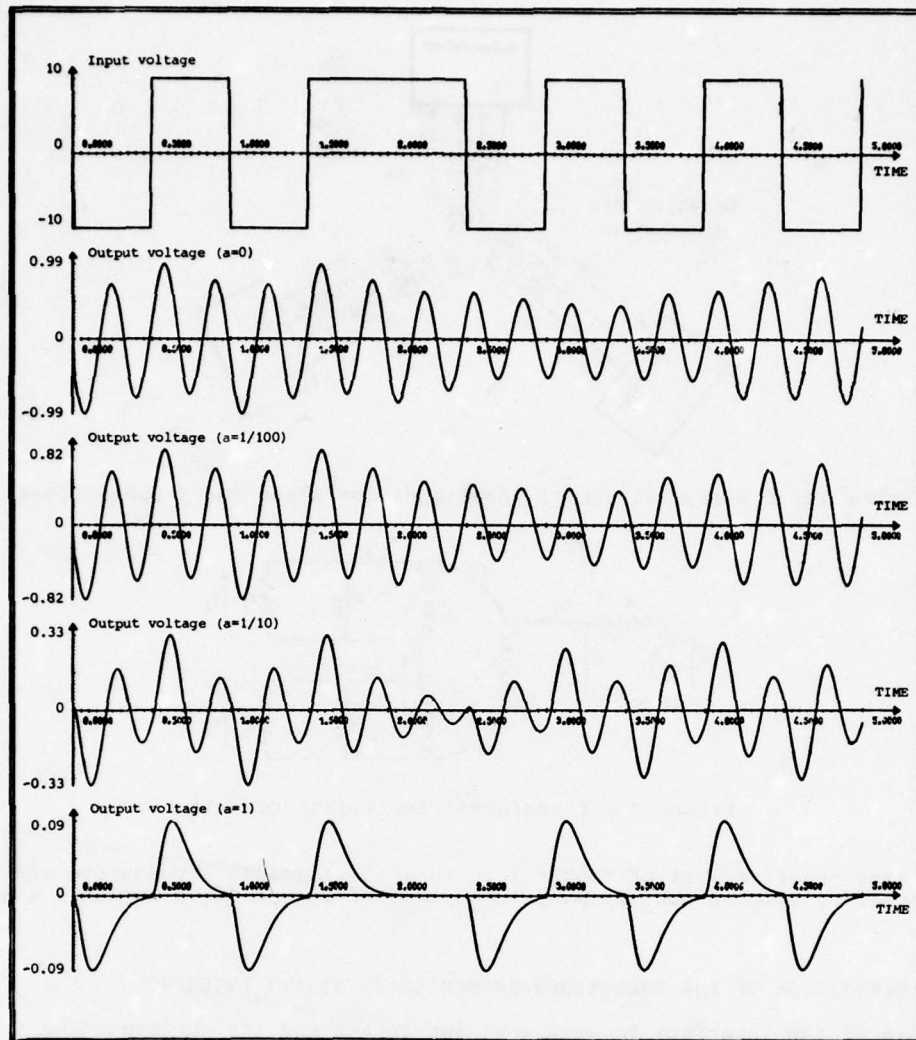


Figure 14: Time response of the loosely-coupled pulse transformer
(coupling coefficient = $1/100$)
 a = series input resistance/output load resistance (R/R')

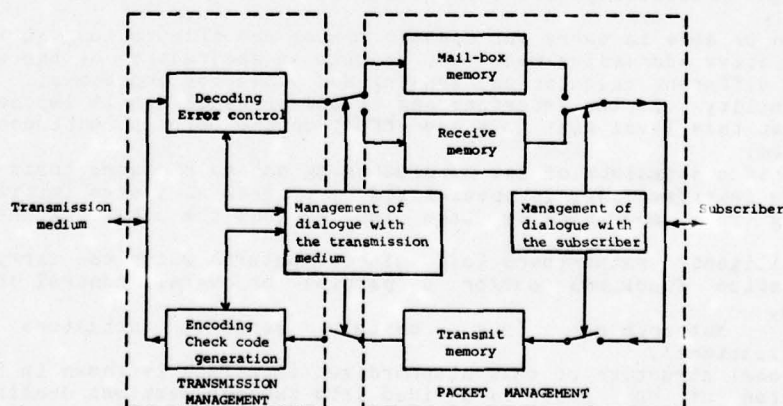


Figure 15: Interface functional structure

III.1.1. Encoding and decoding

The coding technique that has been chosen for the network-structured global transmission medium is a direct consequence of the pulse regenerating technique used in the nodes. The code is necessarily of the monopolar RZ ("return-to-zero") type in which a "1" is coded as a pulse of fixed duration and a "0" is coded as the absence of a pulse in

that bit time.

The functions of the interface at this level thus involve the implementation of a special bit-oriented transmission protocol:

- encoding/decoding of the monopolar RZ code,
- generation and removal of packet delimiting flags,
- bit stuffing/removal in order to obtain message transparency whilst maintaining synchronization and flag uniqueness.

Although a bipolar code would best suit the star-structured local transmission media, it may also be possible to use the same RZ code as in the network transmission medium so as to maintain interface compatibility (cf. curve 5 of figure 12 that shows no overlapping between successive pulses).

III.1.2. Error checking

In order to ensure the safety of the transmission, a 16-bit cyclic redundancy checksum is included in each transmission.

This error-detecting code thus enables fault-tolerance via:

- packet retransmission (for commands),
- use of last non-erroneous value (for broadcasted data).

III.1.3. Access control

The access control functions carried out by the interface are as follows:

- detection of occupation of the transmission medium,
- control of the maximum transmission time (watch-dog function),
- detection of conflicts by a listen-while-talk device,
- execution of the contention access control algorithm.

III.2. Packet management

Packet management is the name given to those functions of the interface that implement the transport level as defined in the introduction to this paper (figure 1).

This level concerns:

- the implementation of a packet transport protocol,
- the execution of buffering functions,

III.2.1. The packet protocol

The packet protocol used in RHEA takes advantage of the use of an associative addressing mode in order to define three basic types of packet exchanges.

III.2.1.1. Data broadcast

This type of exchange consists of the distribution of a data value on the communication medium. Those subscribers interested by this data will recognize its label and store the corresponding information (figure 16).

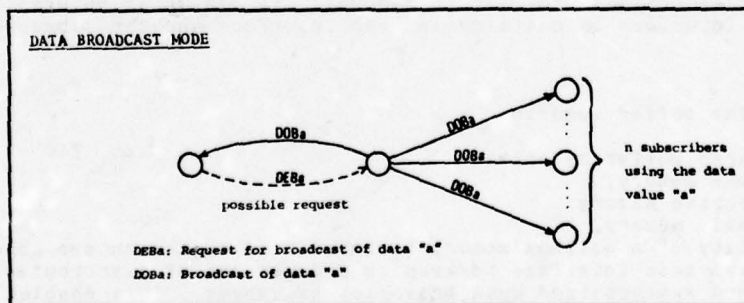


Figure 16

III.2.1.2. Synchronized data broadcast

This type of exchange provides for a simultaneous sampling instant for a set of distributed data that must be synchronized in order for them to have some coherent meaning. A requesting subscriber issues a request with a list of labels corresponding to the set of synchronized data. This request leads to several replies, each corresponding to one of the requested labels (figure 17).

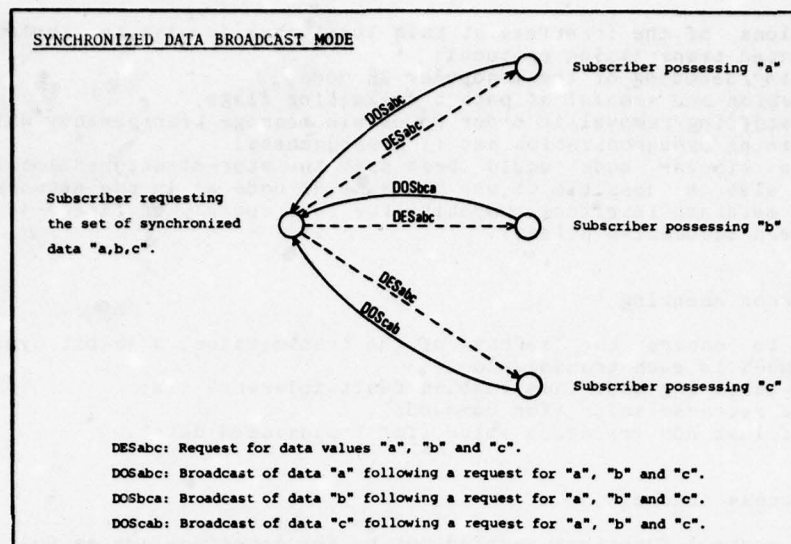


Figure 17

III.2.1.3. Command mode

This type of exchange corresponds to the command mode where the packet has only one destination. The unicity of the destination in this case enables the use of an acknowledgement reply packet in order to ensure a highly secure one-to-one exchange (figure 18).

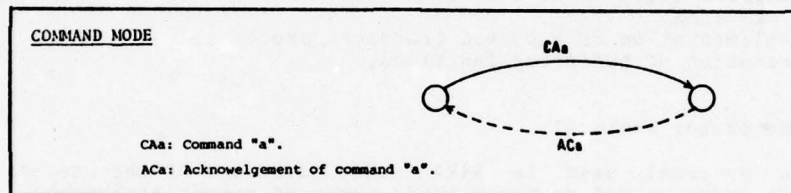


Figure 18

III.2.2. Buffering functions

One of the essential roles of the packet management part of the interface is to provide buffering functions so as to decouple the dialogues between the transmission medium and the interface on one side and the interface and the subscriber on the other side.

III.2.2.1. The buffer memories

There are three buffer memories:

- a mailbox memory,
- FIFO receive memory,
- a transmit memory.

The possibility of a mailbox memory is introduced by the chosen associative address scheme. It enables each interface to keep an updated set of distributed data (using the data broadcast and synchronized data broadcast exchanges). This enables each subscriber to obtain the parameters it requires without burdening the communication system with unnecessary requests.

The FIFO receive memory is used to temporarily store those packets (commands) that must be directly delivered to the subscriber.

The transmit memory is used to store a packet until its successful transmission. In accordance with the chosen contention algorithm, this memory has a capacity of one packet, the subscriber being blocked from further transmissions until the previous packet has been successfully transmitted.

III.2.2.2. Dating

The asynchronous and random nature of the exchanges led us to introduce dating as a means for measuring the amount of time spent by a packet in the transmit and receive memories i.e. the age of a packet when it is effectively consumed.

On receipt of a message/data value from the subscriber, the interface records the "local time" obtained from a real-time clock in the interface. When the corresponding

packet is transmitted, a second reading of the local time enables the "age" of the message/data value to be included in that packet.

A similar procedure is used for measuring the time spent from the moment a packet is received by the destination interface(s) until the corresponding consumption by the destination subscriber(s).

The total "age" of a message/data value is thus known to an accuracy limited by the local clock accuracies and by the propagation time of the transmission medium.

CONCLUSION

In this paper, we have described the definition of the communication support system RHEA intended for the reliable and survivable interconnection of data processing units in an aggressive environment.

The method used for the control of access to the transmission media is contention control which has very good dependability and modularity properties. Further, we have shown that this choice can also be justified from a mean delay viewpoint.

The physical structures of the global and local transmission media have been studied and appropriate solutions defined. The irregular damage-tolerant network used for the global transmission medium uses a very simple node based on a pulse regenerating principle that is suitable for small networks (5-50 nodes) with links in the order of 10m long. The central node of the star structure used for the local transmission media is based on a multi-winding, loosely-coupled pulse transformer.

A standard transmission-medium/subscriber interface has been defined that uses a bit-oriented transmission protocol and an associative addressing packet transport protocol.

A top-down approach has been used throughout with the use of qualitative and quantitative evaluations at each level in order to make motivated choices among the different possible solutions. Work on a prototype system has been started and present research is directed at methods of achieving fault recovery and dynamic system reconfiguration.

BIBLIOGRAPHY

- [1]...D.R.POWELL, J.C.LAPRIE, P.ROMAND, G.ALCOUFFE: "A reliable and survivable data transmission system for avionics processing", Proceedings of the AGARD-GCP Symposium on the Impact of Integrated Guidance and Control Technology on Weapons Systems Design, Sandefjord, Norway, 9-12 May 1978, pp. 22/1-22/12.
- [2]...D.POWELL, J.C.LAPRIE, P.ROMAND, C.ALEONARD: "RHEA: A system for reliable and survivable interconnection of real-time processing elements", Proceedings of the IEEE 8th. International Fault-Tolerant Computing Symposium (FTCS-8), Toulouse, France, 21-23 June 1978, pp. 117-122.
- [3]...K.J.THURBER, E.DOUGLAS JENSEN, L.A.JACK, L.L.KINNEY, P.C.PATTON, L.C.ANDERSON: "A systematic approach to the design of digital bussing structures", Proceedings of the AFIPS Fall Joint Computing Conference, Anaheim, California, USA, 5-7 Dec., 1972, pp. 719-740.
- [4]...E.DOUGLAS JENSEN: "The Honeywell experimental distributed processor - an overview", Computer, vol.11, no.1, Jan. 1978.
- [5]...R.M.METCALFE, D.R.BOGGS: "Ethernet: distributed packet switching for local computer networks", Communications of the ACM, vol.19, no.7, July 1976.
- [6]...R.SOMMER: "COBUS, a firmware controlled data transmission system", Proceedings of the Second Symposium on Microprocessing and Microprogramming, Venice, Italy, 14-16 Oct. 1976, EUROMICRO, North-Holland Publishing Company, 1976, pp. 299-304.
- [7]...C.ALEONARD, M.BUIS, J.P.DESMOULINS, J.C.LAPRIE, J.MARCO, D.POWELL, P.ROMAND: "Recherche de la sécurité dans les réseaux de transmission numérique. Rapport final", Contrat DRET no.76/274, LAAS Publication no.1733, Toulouse, France, June 1978.
- [8]...G.A.ANDERSON, E.DOUGLAS JENSEN: "Computer interconnection structures: taxonomy, characteristics and examples", Computing Surveys, vol.7, no.4, Dec. 1975.
- [9]...T.BA3IL SMITH III: "A damage and fault-tolerant input/output network", IEEE Transactions on Computers, vol.C-24, no.5, May 1975.
- [10]...W.GREENE: "A review of classification schemes for computer communication networks", Computer, vol.10, no.11, Nov. 1977.
- [11]...F.K.HANNA, O.R.HINTON, K.R. DIMOND, T.ACIAK, P.J.MYERS, C.DAWSON: "A microprocessor network for distributed industrial control", Proceedings of the IEE Distributed Computer Control Conference, Birmingham, England, 26-28 Sep. 1977, pp. 70-73.

ACKNOWLEDGEMENTS

This work was carried out under contract nos. 76/274 and 73/352 from the "Département des Recherches, Etudes et Techniques" and in collaboration with "Société CROUZET". Our thanks must go to Messrs. CAVARROC, DESMOULINS and ROMAND from CROUZET, and to Mr. BUCHWALTER from LAAS for their useful discussion and help in producing this paper.

DEVELOPMENT OF AIDING GPS/STRAPDOWN INERTIAL NAVIGATION SYSTEM

by

Dr. D.F. Liang

Defence Research Establishment Ottawa, Ottawa, Canada

Dr. D.B. Reid

Philip A Lapp Ltd., 14A Hazelton Ave., Toronto, Canada

R.H. Johnson

S and S Software Ltd., 444 MacLaren St., Ottawa, Canada

and

B.G. Fletcher

Defence Research Establishment Ottawa, Ottawa, Canada

SUMMARY

This paper presents an overview of the design and development of an integrated multisensor navigation system comprised of a NAVSTAR GPS receiver, an aiding strapdown inertial navigation system (ASIN) and a number of auxiliary sensors, namely, air data and strapdown magnetic sensors.

In the present phase, comprehensive software packages have been developed to simulate all the subsystems used. A modular and computationally efficient Kalman filtering algorithm was designed and implemented for the integration of the GPS and ASIN. During the course of the development, two new techniques have been developed. An exact algorithm was derived to transform inertially referenced data into geographic coordinates. Also a dual channel attitude algorithm has been formulated which increases the bandwidth of the attitude computation in the strapdown navigator.

Other routines developed include the baro-damping algorithm, auxiliary sensor processing and calibration routines. To provide a baseline level of performance, simulation results have been obtained for future flight testing of the hardware.

1.0 INTRODUCTION

The NAVSTAR Global Positioning System (GPS) is a 24 satellite navigation system that will provide highly accurate timing and three dimensional position and velocity information to suitably equipped military and civilian users. The GPS, being a radio-ranging system, will provide continuous worldwide navigation information to an unlimited number of users, regardless of weather conditions.

Although GPS represents a significant advance in navigation technology, its dependence on electromagnetic radiation from external sources (satellites) makes it susceptible to interference and jamming. The degradation of navigation accuracy is most pronounced when the GPS signal-to-noise ratio is low and the vehicle is undergoing highly dynamic maneuvers.

Jamming resistance and signal tracking performance of the GPS receiver can be enhanced by aiding of the receiver's code and carrier loops with velocity data from an aiding inertial navigation system (INS). Additional benefits which can be realized by GPS/INS integration include:

- continual provision of navigation information during periods of GPS outage due to jamming or receiver failure
- in-flight calibration and alignment of the inertial navigator
- rapid acquisition and reacquisition of the GPS signals
- GPS antenna steering using INS attitude and angular rate data.

Recent advances in gyro technology and low-cost high-speed mini-computers have greatly accelerated the acceptance of strapdown INS as a serious contender for general navigation and avionic usage. The overall costs of ownership and maintenance are expected to be lower than the conventional gimballed INS, without any sacrifice in the level of navigation accuracy. One further incentive is the ability of the highly accurate GPS to enhance considerably the performance of a low-cost low-quality inertial measurement unit (IMU) through the calibration of its sensors.

In view of the above considerations, DREO has undertaken the design and development of an aiding strapdown inertial navigation (ASIN) system, to be integrated with the GPS receiver presently under development at Canadian Marconi.

In this paper, the overall integrated GPS/ASIN system concept is presented and performance characteristics of the integrated system are evaluated using computer simulations. New algorithms developed by DREO and contractor personnel [1] are also presented.

2.0 System Overview and Subsystem Model Simulations

The structure of the hybrid GPS/ASIN system is illustrated in Figure 1.

The IMU employed is a low accuracy Honeywell strapdown H478F package consisting of 3 GG1111 SDF floated gyros and 3 Q-Flex pendulous accelerometers. Auxiliary heading data are provided by a triad of magnetometers strapped to the airframe, which measures the three body axis components of the earth's magnetic field. Air data sensors are employed to provide airspeed and barometric altitude for Schuler loop and vertical channel damping.

System alignment and integration functions are performed by a high-order extended Kalman filter and a feedback error-control algorithm. The filter combines GPS range and range-rate data with ASIN position and velocity information, and from these data, estimates of the major time-correlated system errors and error sources are generated. The filter's error estimates are fed back to the navigator, where they are used to reset computed attitude, position and velocity as well as null acceleration and angular rate error inputs.

The most recent phase of the DREO GPS/ASIN project concentrated on the development of comprehensive software packages for simulation of the hybrid system. Figure 2 illustrates the structure of this package, showing the inter-relationship of the primary component routines. The GPS/ASIN sub system model simulation is driven by PROFGEN, an aircraft profile generation program developed by the Air Force Avionics Laboratory, WPAFB. The program was used to provide reference position, velocity, acceleration, attitude and angular rate data consistent with flight trajectories of a Convair 580*. These data are applied as inputs to the GPS, inertial sensor and physical parameter simulation routines and provide a baseline for evaluation of the ASIN and hybrid GPS/ASIN navigation outputs. The other sub system model simulations are:

A) Inertial Sensor Error Model Simulation:

Comprehensive error models of the accelerometer and gyro triads have been simulated. The deterministic models are similar in part to those of D. Dove [2]. The most significant differences are the addition of temperature sensitive errors and the method used to estimate the average dynamic misalignment of the gyro input axis. The error parameters of the sensor models are perturbed initially from those parameters used in the compensation model. This perturbation was carried out using the mean and 1 σ values as specified in the H478F specs.

B) Air Data Error Model Simulation:

The air data error models comprise three components: static pressure errors, total pressure errors and temperature errors. In the physical parameter simulation program, pressure gradient effects, as well as temperature and pressure variations have been simulated. The model also includes the effect of passing through weather fronts.

C) Magnetometer Error Model Simulation:

The outputs of the strapdown magnetic sensors are corrupted with both low-frequency and broad-band errors which, without compensation, can produce large errors in computed heading.

D) GPS Error Model Simulation:

A comprehensive software program has been developed to simulate all the significant error sources of the GPS. These errors include ephemeris errors, tropospheric and ionospheric time delay errors, satellite and receiver clock errors. Other sources of error are the quantization and computational errors. The details of the GPS error model adopted for integration with the strapdown INS are presented in Section 4.

E) GPS Simulation:

A software program has been developed for the full (24) satellite constellation. The GPS receiver simulator [3] generates ranges and range rates to all visible satellites and selects the best geometry that minimizes the GDOP (geometric dilution of precision). Exact values are perturbed using GPS error models and compensation was performed so as to reduce systematic error contributions.

3.0 Mechanization Details

3.1 Navigation Mechanization

In the present ASIN development a 3rd order quaternion attitude algorithm iterated at 50 Hz was selected over the more classical direction cosine approach, to transform the body frame data to the inertial frame, due to its greater efficiency and accuracy. The strapdown navigation equations have also been implemented in the inertial frame, as opposed to the local-level frame used by gimbaled systems since, for strapdown systems, body angular rates are measured by the gyros with respect to inertial space. The detailed mechanization of the computational algorithm is presented in Figure 3.

As shown in Figure 3, a unique technique has been developed at DREO to accurately compute latitude and altitude from the inertial frame coordinates. The derivation, as given in Appendix I, is exact and conceptually simple. The altitude and latitude at any point P near the earth can be obtained using the geometric relationship that the distance from P to its projection Q on the reference ellipsoid is the minimum distance from P to the ellipsoid.

*The Convair 580 experimental flight laboratory developed and operated by the Canadian National Aeronautics Establishment will be used for future flight tests of system hardware.

3.2 Air Data Algorithms

The air data algorithms implemented to compute true air speed and barometric altitude from sensed atmospheric temperature and static and total pressure have been obtained from the literature [4,5]. The airspeed algorithm is conventional and similar to that mechanized in most air data systems. The altitude algorithm, due to Blanchard [5] is an improvement over conventional algorithms in that it utilizes atmospheric temperature data in addition to sensed static pressure to compute pressure altitude. This improves the accuracy of the altitude computation and, in particular, effectively eliminates the altitude scale factor error present in conventional mechanizations which assume a standard atmosphere temperature profile.

3.3 Altitude Damping

The altitude and vertical velocity computations in an unaided inertial system are unstable, producing errors which grow exponentially with time. In the DREO application, the inertial vertical channel is stabilized using altitude data obtained from a barometric altimeter.

A third-order baro-damping algorithm, derived in part from Reference 6, has been designed specifically for inertial systems which employ the inertial coordinate frame for navigation computation. As shown in Figure 4, computed altitude is differenced with baro altitude and the residual δh is fed back into the vertical channel through three damping loops. The damping loop gains have been chosen using purely analytical optimization methods to satisfy the following criteria:

- a) provide stable, non-oscillatory loop behaviour,
- b) minimize the RMS altitude error with respect to broad-band acceleration and baro-altimeter disturbances, and
- c) minimize transient altitude errors due to step-like baro error inputs, given condition b) above.

In order to obtain a purely analytical solution, the expressions for gain selection were formulated with the constraint that two vertical channel poles be located at $-1/T$ and the third pole at $-1/NT$, where T is the system time constant and N can be any positive non-zero real number. It is shown in [1] that, to satisfy criterion b), N and T should be chosen as

$$T = \left[\frac{(5N^3 + 18N^2 + 14N + 2)}{3N^3} \frac{q_b}{q_a} \right]^{1/4}$$

and $N > 0.01$ (for assumed values of q_b and q_a)

where q_a and q_b are the spectral densities of the broad-band acceleration and baro errors respectively. In addition, it is shown that in order to satisfy criterion c), N must be chosen much less than unity ($0 < N \ll 1$).

For assumed transport aircraft flight conditions and sensor error model parameters, typical values of q_a and q_b are estimated to be

$$q_a \leq 1.2 \times 10^{-2} \text{ (ft/s}^2\text{)}^2/\text{Hz}$$

$$q_b = 6250 \text{ ft}^2/\text{Hz}$$

which, for $N = .05$, yields an optimum time constant of

$$T = 250 \text{ seconds}$$

The corresponding gain set places two (slow) system poles at $-1/250 \text{ (sec}^{-1}\text{)}$ and one (fast) pole at $-1/12.5 \text{ (sec}^{-1}\text{)}$. Simulation results presented in [1] show that this gain set provides excellent damping performance. For the assumed simulation conditions, the RMS vertical velocity error was found to be typically less than 2 ft/sec. The damping algorithm provided significant attenuation of broad-band baro disturbances in the altitude computation but forced computed altitude to track baro-altitude over the long term.

3.4 Dual Channel Attitude Algorithm

A new dual-channel attitude algorithm was formulated to increase the bandwidth of the attitude computation in the strapdown navigator [1]. As shown in Figure 5, the algorithm splits the incoming angular rates from the strapdown gyros into high and low frequency components and processes these components separately in high and low rate channels.

This algorithm was originally derived to reduce attitude and position error drifts resulting from high frequency (vibrational) coning motion inputs. However, the algorithm is in fact quite general and could be employed to improve attitude performance irrespective of the vibrational environment of the navigator.

Simulation results show that with the standard third-order 50 Hz quaternion attitude algorithm, a z-axis coning "drift" of about 2×10^{-4} rad/sec induced by x and y-axis vibrational rates of 20 Hz (90° out-of-phase) and coning half-angle of 0.1 degree, will produce heading and radial position error drifts of approximately 0.007 deg/sec and 40 meters/sec respectively. When the dual channel algorithm is employed with low and high channel iteration rates of 50 to 500 Hz respectively, the corresponding system errors

were reduced by more than a factor of 20 for the same coning motion inputs while all inertial sensor error models were disabled. However, it should be noted that when the inertial sensor error models were activated and coning rates applied an interaction between the coning motion simulation and inertial error models resulted in position errors of extremely large magnitudes. Coning compensation was found to improve position accuracy and yield heading error drift rates consistent with earlier runs. However, the radial position error remained larger than expected. It therefore appears that vibrational inputs excite some significant error modes in the system sensors (probably accelerometers) which cannot be entirely compensated for by increasing the bandwidth of the attitude algorithm. In light of the above, the exact nature of the interaction between the coning motion and inertial error simulations warrants further investigation. The best choice of certain algorithm parameters (high and low channel iteration rates, low-pass filter order and cut-off frequency, etc) and the potential computer loading problem, which could be encountered in real-time application, must also be further investigated.

4. GPS/ASIN INTEGRATION

The GPS/ASIN integration function is performed by a 49-state Kalman filter and feedback error control algorithm (Figure 6). GPS measured ranges (ρ_{GPS}) and range-rates ($\dot{\rho}_{GPS}$) from the receiver to four satellites are differenced with ranges (ρ_{INS}) and range rates ($\dot{\rho}_{INS}$) computed by combining inertially indicated position and GPS satellite ephemeris data to provide a set of eight error observations. These observations (z) are processed recursively by the filter to compute optimum estimates (\hat{x}) of the ASIN and GPS error states (x). The estimates are obtained by utilizing knowledge of the statistics of system and measurement errors, and linearized models of the system and measurement error dynamics. The filter's error estimates are fed back into the navigator via the error control algorithm which instantaneously corrects the position, velocity and attitude integrators and continuously compensates the accelerometer and gyro outputs for estimated acceleration and angular-rate error effects.

The basic structure of the GPS/ASIN error model and salient features of the filter and error control mechanization [1] are reviewed below.

4.1 GPS/ASIN Error Model

The GPS/ASIN system error model mechanized by the integration filter is of the form:

$$\dot{\underline{x}}(t) = \begin{bmatrix} \dot{\underline{x}}_{INS} \\ \dot{\underline{x}}_{GPS} \end{bmatrix} = \begin{bmatrix} \underline{F}_{INS}(t) & 0 \\ 0 & \underline{F}_{GPS}(t) \end{bmatrix} \begin{bmatrix} \underline{x}_{INS}(t) \\ \underline{x}_{GPS}(t) \end{bmatrix} + \begin{bmatrix} \underline{G}_{INS}(t) & 0 \\ 0 & \underline{G}_{GPS}(t) \end{bmatrix} \begin{bmatrix} \underline{u}_{INS}(t) \\ \underline{u}_{GPS}(t) \end{bmatrix}$$

$$\underline{z}(t_k) = [\underline{A}_{INS}(t_k), \underline{A}_{GPS}(t_k)] \begin{bmatrix} \underline{x}_{INS}(t_k) \\ \underline{x}_{GPS}(t_k) \end{bmatrix} + \underline{v}_{GPS}(t_k)$$

where

$\underline{x}_{INS}, \underline{x}_{GPS}$ = INS, GPS error state vectors estimated by the Kalman filter

$\underline{F}_{INS}, \underline{F}_{GPS}$ = INS, GPS error dynamics matrices

$\underline{u}_{INS}, \underline{u}_{GPS}$ = INS, GPS white process noise vectors with spectral densities Q_{INS}, Q_{GPS} respectively.

$\underline{G}_{INS}, \underline{G}_{GPS}$ = INS, GPS noise input matrices.

\underline{z} = range and range-rate error observation vector

$\underline{A}_{INS}, \underline{A}_{GPS}$ = INS, GPS error measurement matrices

\underline{v}_{GPS} = GPS white measurement noise vector with covariance R_{GPS}

The ASIN error state \underline{x}_{INS} is a 42-element vector containing the major time-correlated errors and error sources of the inertial system (see Table 1). These are:

- inertial position errors (3 states)
- inertial velocity errors (3)
- platform misalignment angles (3)
- acceleration error sources (15)
- angular rate error sources (18)

	Variable Number	Symbol	Type	Definition	Initial value ($\sqrt{P_{11}}$)	Correlation time (τ_{11}) (sec)	Process Noise Spectral Density	Units for x_1 and $\sqrt{P_{11}}$
x_{SYS}	1-3	$\delta \underline{r}$		position error vector	10	-	$\neq 0$	meters
	4-6	$\delta \underline{v}$		velocity error vector	.05	-	$\neq 0$	m/sec
	7	$\delta \underline{a}$		vertical acceleration bias	2.5×10^{-3}	-	$\neq 0$	m/sec
	8-10	$\underline{\psi}$		INS misalignment angles	9×10^{-3}	-	$\neq 0$	radians
x_{RC}	1	δb	EC	bias altitude error	200	$\frac{5 \times 10^{-5}}{V_g}$	$\frac{2P_{11}}{\tau_{11}}$	meters
	2-4	δg^s	EC	vertical deflections & gravity anomalies	$.245 \times 10^3$	$\frac{4 \times 10^{-4}}{V_g}$	$\frac{2P_{11}}{\tau_{11}}$	m/sec ²
x_{RW}	1-3	\underline{a}^b	RW	accelerometer bias	1.47×10^{-3}	-	1.2×10^{-12}	m/sec ²
	4-6	\underline{c}^b	RW	gyro bias	5×10^{-6}	-	2.72×10^{-16}	rad/sec
x_{RC}	1-3	δa^b	RC	accelerometer scale factor error	1.5×10^{-4}	-	0	
	4-10	\underline{a}^b	RC	accelerometer axes misalignment	5×10^{-4}	-	0	radians
	11-13	δa^b_g	RC	gyro scale factor error	2×10^{-4}	-	0	
	14-20	\underline{a}^b_g	RC	gyro axes misalignment errors	7.0×10^{-4}	-	0	radians
	21-24	\underline{c}^b_g	RC	g-sensitive drift errors	1.2×10^{-6}	-	0	(rad/sec) (m/sec ²)

TABLE 1 - ESTIMATED STRAPDOWN INERTIAL NAVIGATOR ERROR MODEL PARAMETER VALUES FOR H478F

The estimated acceleration and angular rate error sources consist of (a) sensor bias instability errors modelled as random walks (RW), (b) sensor scale factor errors, axes misalignments and g-sensitive gyro drifts modelled as random constants (RC), and (c) residual broad-band errors modelled as white noise. In addition, acceleration errors due to vertical deflections and gravity anomalies are modelled as exponentially distance-correlated random processes (EC).

The GPS error state x_{GPS} contains 7 time-correlated GPS error sources (see table 2), namely the user clock time offset, frequency offset, frequency drift rate and four range errors.

	Variable Number	Symbol	Type	Definition	Initial Value	Correlation time (τ_{11}) (sec)	Process Noise Spectral Density	Units for x_1 and $\sqrt{P_{11}}$
x_{GPS}	1	τ		time error of user clock	1×10^{-6}	-	0	sec
	2	$\overline{\tau}$	RW	(inverse) frequency offset error of user clock	1×10^{-9}	-	10^{-20}	sec/sec
	3	\underline{d}	RW	(inverse) frequency drift rate error of user clock	1×10^{-10}	-	10^{-20}	sec/sec ²
	4-7	$\delta \underline{p}$	EC	clock range error	3	$\frac{2.8 \times 10^{-4}}{V_R}$		m

TABLE 2 - ESTIMATED GPS ERROR MODEL PARAMETER VALUES

4.2 U-D Filter and Error Control Mechanization

The filtering algorithm which has been implemented at DREO is based on the 'U-D' covariance factorization method [7]. The U-D mechanization has the numerical accuracy and stability characteristics of square root Kalman filters but is considerably more efficient. Scalar sequential observation processing is used resulting in increased efficiency and flexibility, allowing for simple processing of irregular observations and easy rejection of erroneous measurement data.

In the U-D mechanization, the filter's estimation error covariance matrix P is expressed in terms of its factors

$$P = UDU^T$$

where D is diagonal and U is an upper triangular matrix with unit diagonal. Covariance and state updates at observation times and extrapolations between updates are carried out entirely in terms of U-D factors so that the covariance matrix P is never explicitly computed.

The extrapolation of the U-D factors between updates is performed using Bierman's modified weighted Gram-Schmidt (MWG-S) orthogonalization algorithm [7].

The filter software developed for the DREO application has been structured to offer maximum

flexibility in data processing, and, at the same time, to maintain modularity of different program sections. The linearized models of system and measurement error dynamics and statistics are modelled by the continuous, time-variable system equations. The corresponding discrete-time transfer functions and noise covariances required by the filter are computed at each time point using power series techniques in which matrix block structure is utilized to improve efficiency. The user may select (almost) any subset of the error state variables for estimation, and the corresponding system matrices are constructed automatically by the program. This feature will prove extremely useful in the future when reduced-order suboptimal filter designs are to be implemented and evaluated.

The program modules required for computation of the error model matrices have been separated from the U-D update and MWG-S extrapolation programs making the software package completely general.

The error control strategy employed for GPS/ASIN integration is illustrated in Figure [7]. Both continuous and impulsive correction techniques are used. Specifically, ASIN attitude, velocity and position parameters are reset impulsively following each filter update cycle to null estimated system errors. Acceleration and angular rate correction vectors ($\Delta \mathbf{a}^b$ and $\Delta \boldsymbol{\omega}^b$) are computed continuously using models of the sensor error processes and are subtracted from the sensor outputs before processing by the navigator. The sensor error models employed for error control are duplicates of the models imbedded in the Kalman filter, and are updated impulsively following each filter update cycle. After all control inputs are applied, the corresponding error estimates are zeroed in the filter and remain zero until the next filter update time.

The closed-loop GPS/ASIN mechanization keeps inertial system error magnitudes small and so maintains the validity of the linearity assumptions on which the Kalman filter's error models are based. The mechanization also tends to reduce the system's sensitivity to unmodelled disturbance inputs.

5.0 SIMULATION RESULTS

Computer simulations of the hybrid GPS/ASIN system were performed to evaluate the Kalman filter and feedback error control algorithms. The GPS/ASIN simulations were driven by a 1750 second race track flight profile as shown in Figure 8.

The simulated GPS receiver generated average total position and velocity errors of 24.7 meters and 0.541 meters per second respectively. The receiver included ephemeris errors, quantization errors and user clock errors. The inertial navigator simulation included inertial sensor error sources, gravity anomalies and vertical deflections, and initial condition errors. The Kalman filter used the above GPS and INS data to perform updates at one second intervals.

To evaluate the performance of the hybrid system, the following three different cases were considered:

- a) a 45 - state filter with 42 inertial error states and 3 GPS clock error states
- b) a 49 - state filter with 45 GPS and INS states plus four additional time correlated GPS range error states
- c) a 47 - state filter with 45 GPS and INS states plus 2 additional altitude damping error states.

Simulation results for the above three cases in terms of average and RMS position and velocity errors are presented in Figures 9-14 and Table 3. Figures 15-16 are simulation results using the 45 state filter operating on GPS data containing no ephemeris, quantization or user clock errors, but perturbed by pseudo-range and pseudo-range rate additive white noise processes of 10m and 0.2m/s, respectively.

Filter Configuration	Total (RSS) Position			Total (RSS) Velocity		
	Average Error	RMS Error	RMS Bound	Average Error	RMS Error	RMS Bound
45 states	6.16	7.08	3.89	0.0966	0.106	0.112
47 states (with baro damping)	6.72	7.89	5.88	0.101	0.111	0.117
49 states (with range error states)	5.66	6.42	7.34	0.0955	0.105	0.114

TABLE 3 - GPS/ASIN SYSTEM POSITION AND VELOCITY PERFORMANCE

In the case of the 45-state filter, simulation results are presented in Figures 17-19 for x-axis misalignment error, acceleration and angular rate errors. These results are typical of all three K. Filters considered. It is noted that platform misalignment errors, which at run time were 3 to 4 degrees, reduced to less than 100 arc second (RMS) per axis toward the end of the simulation runs. In all the above figures, solid lines represent actual errors and the dashed lines represent the Kalman filter indicated RMS error bounds (square roots of the appropriate error covariances).

It can also be seen that the system position and velocity errors of Figures 9-10 are significantly higher than those of Figures 15-16, which is solely due to the satellite ephemeris errors of the GPS receivers. To compensate for the ephemeris error effects, four GPS range error states were added to the 45-state filter. Amongst the three primary cases considered, the 49-state filter with results presented in Figures 11-12, was judged to be the best of the three. The 49-state filter effectively reduced the average RSS position error of 45-state filter by about 0.5 meters, and the indicated RMS bounds appeared more consistent with the actual system performance.

To investigate the performance of the hybrid system with altitude damping, two baro-damping filter states were added. The results presented in Figures 13-14, indicate that while good navigation performance is still attained, position errors are significantly noisier than the case without baro-damping. The degradation is due to the insertion of broad-band altimeter noise into the system. This result suggests that the inclusion of baro-damping is not advantageous when GPS information is available.

6.0 CONCLUSIONS

The objective of this paper was to outline the design and development of a GPS/ASIN integrated navigation system. In the most recent phase of this project, a comprehensive software simulation package has been developed to simulate, in detail, all subsystems which make up the hybrid GPS/ASIN system. The package can be used to establish a baseline of achievable results for future flight tests of system hardware. Preliminary simulation results demonstrate very good performance characteristics of the integrated system. Average position and velocity errors of about 6m and 0.1 m/s, and component steady-state misalignments of less than 150 arcsec were observed.

However, it should be noted that additional work must be carried out in order to validate all error model parameters implemented. This will be performed during flight trials on the NAE Convair 580. Moreover, the high order filter mechanizations developed to date are not amenable for implementation on a real-time computer. Therefore, reduction of the state-size will be required, using the covariance analysis technique to eliminate secondary error sources.

7.0 REFERENCES

1. Reid, D.B., McWilliam, B.N., Gesing, W.S. "Simulation Study Of A Hybrid GPS/Strapdown Inertial Navigation System" Report No. 3/F/79, Philip A. Lapp Ltd., March 1979.
2. Dove, D.W., "A Performance Evaluation Of A Strapdown Inertial Measurement Unit In The Presence Of A Severe Dynamic Environment", M.I.T. Report T-544, Jan. 1971.
3. Dymont, M., "Global Positioning System Simulation", DREO Unpublished Report, Sept. 1978.
4. Kayton, M. and Fried, W.R., ed., Avionics Navigation Systems, John Wiley and Sons, 1969.
5. Blanchard, R.L., "A New Algorithm For Computing Inertial Altitude And Vertical Velocity", IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-7, No. 6, November, 1971.
6. Nash, R.A. Jr., and Hutchinson, C.E., "Altitude Damping Of Space-Stable Inertial Navigation Systems", IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-9, No. 1, Jan. 1973.
7. Bierman, G.J., Factorization Methods for Discrete Sequential Estimation, Academic Press, 1977.

ACKNOWLEDGEMENT

The authors wish to thank Mr. K.A. Peebles of DREO for his leadership in the project. Thanks are also due to Dr. W.S. Gesing and Mr. B.N. McWilliam of Philip A. Lapp Ltd., and Mr. L.J. Engel of DREO for their valuable contributions to the work presented in this paper.

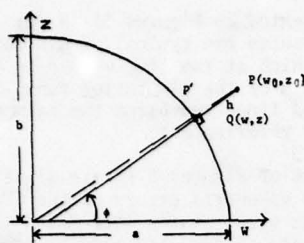


Figure 20

APPENDIX 1

LATITUDE - ALTITUDE ALGORITHM

Suppose we are given an arbitrary point $P(x_0, y_0, z_0)$ above the reference ellipsoid in inertial frame coordinates. Taking the plane containing P and the polar axis Z as in figure 20, then $w_0 = \sqrt{x_0^2 + y_0^2}$.

The altitude h and geographic latitude ϕ of P can be obtained by solving for the coordinates of the point $Q(w, z)$ on the ellipse directly beneath P .

$$\text{Now } h^2 = (w - w_0)^2 + (z - z_0)^2 \quad (\text{distance equation}) \quad (1)$$

$$\text{and } w = \sqrt{\frac{a^2 b^2 - a^2 z^2}{b^2}} \quad (\text{ellipse equation}) \quad (2)$$

Substituting Eq.(2) into Eq.(1) we obtain the (distance)² from P to Q :

$$h^2 = a^2 \left(1 - \frac{z^2}{b^2}\right) - 2w_0 \sqrt{a^2 \left(1 - \frac{z^2}{b^2}\right)} + w_0^2 + z^2 - 2z_0 z + z_0^2 \quad (3)$$

Note that Q is the point on the ellipse with the minimum distance from P .

Differentiating Eq.(3) w.r.t. z and setting it to zero yields:

$$\sqrt{1 - \frac{z^2}{b^2}} = \frac{-w_0 a z}{b^2(z - z_0) - a^2 z} \quad (4)$$

The following quartic expression results after squaring both sides of Eq.(4):

$$\left[\frac{(b^2 - a^2)^2}{b^2} z^4 + \frac{2(b^2 - a^2)}{b^2} z_0 z^3 + \left[\frac{(b^2 - a^2)^2}{b^2} - z_0^2 - \frac{w_0^2 a^2}{b^2} \right] z^2 - 2(b^2 - a^2) z_0 z + b^2 z_0^2 \right] = 0 \quad (5)$$

Hence z can be solved for exactly using the standard quartic formula or it can be found using the Newton-Raphson iterative technique. Using the iterative method and a starting value of

$$z = \frac{b}{\sqrt{1 + \left(\frac{b w_0}{a z_0}\right)^2}}$$

corresponding to the point P' in figure 20, it was found that only 2 or 3 iterations are necessary to find z . Starting with Eq.(2) and then using Eq.(1) we can solve for the altitude h of the point P .

Note that since P and Q have the same geographic latitude ϕ , the latitude of P is obtained using the point Q :

$$\phi = \tan^{-1} \left(\frac{a^2 z}{b^2 w} \right)$$

Further, for any point P inside the ellipse, quartic Eq.(5) also generates distance-to-ellipse extrema and thus it can be seen geometrically that for depths $\leq \frac{1}{2}$ earth radius, the Newton-Raphson technique will find the proper (minimum-distance-producing) root of Eq.(5).

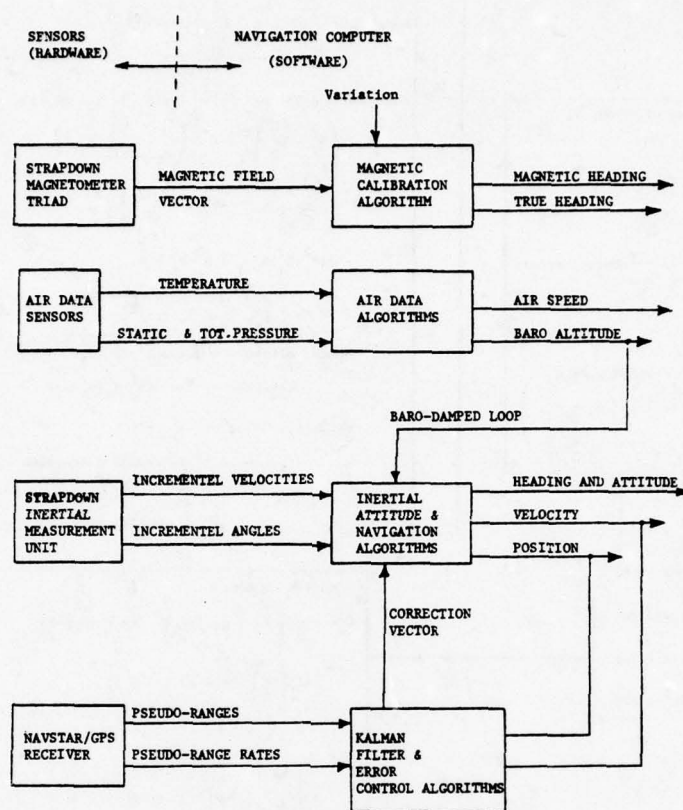


FIGURE 1 HYBRID GPS/ASIN CONFIGURATION

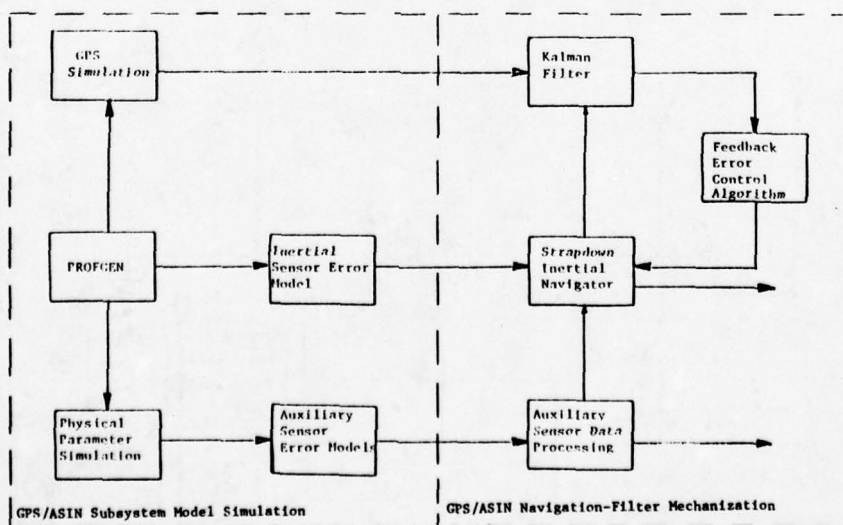


FIGURE 2 BASIC STRUCTURE OF GPS/ASIN SIMULATION SOFTWARE

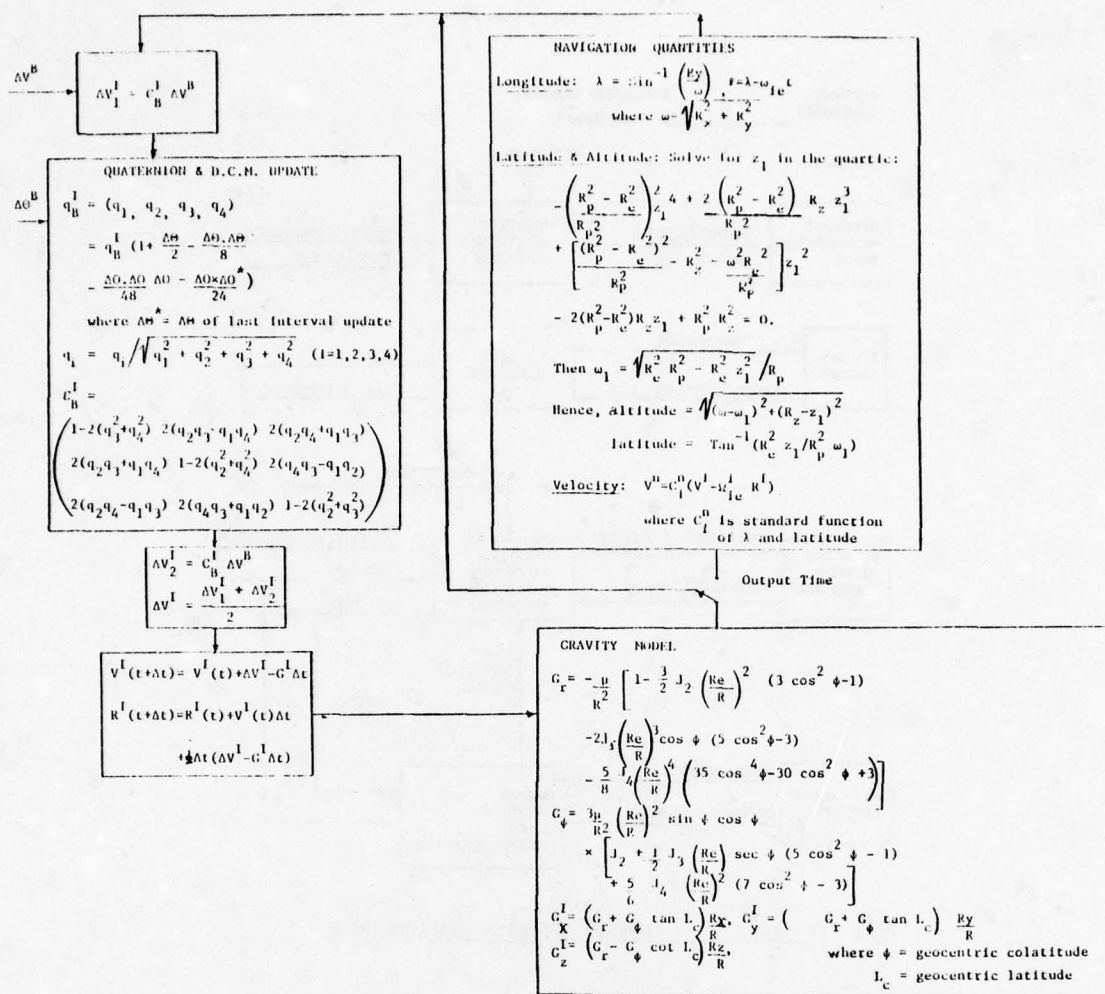


FIGURE 3 STRAPDOWN INERTIAL NAVIGATION MECHANIZATION

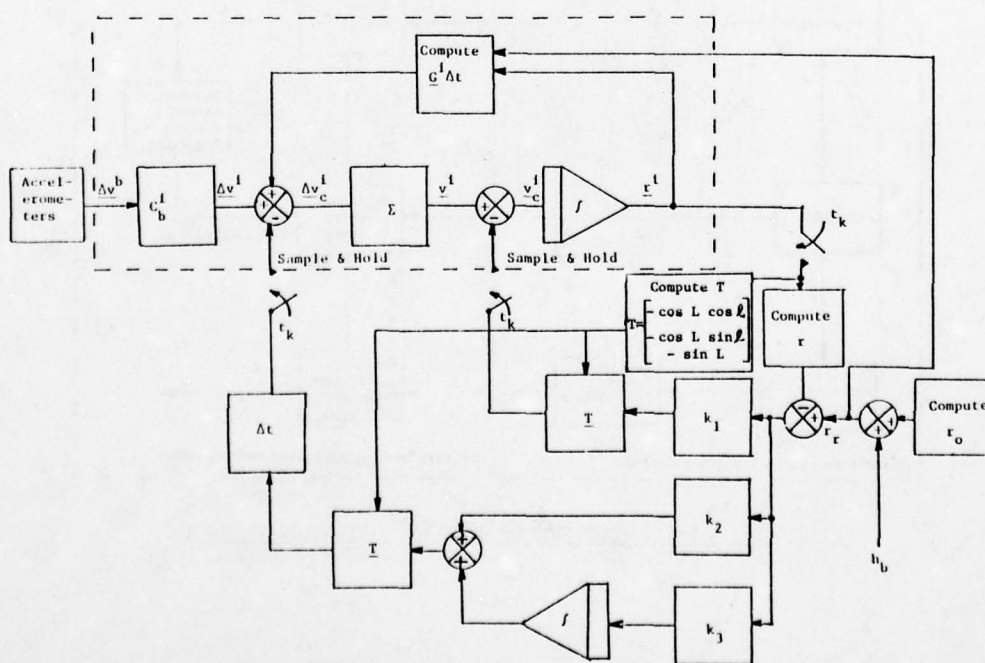


FIGURE 4 BARO/INERTIAL DAMPING MECHANIZATION

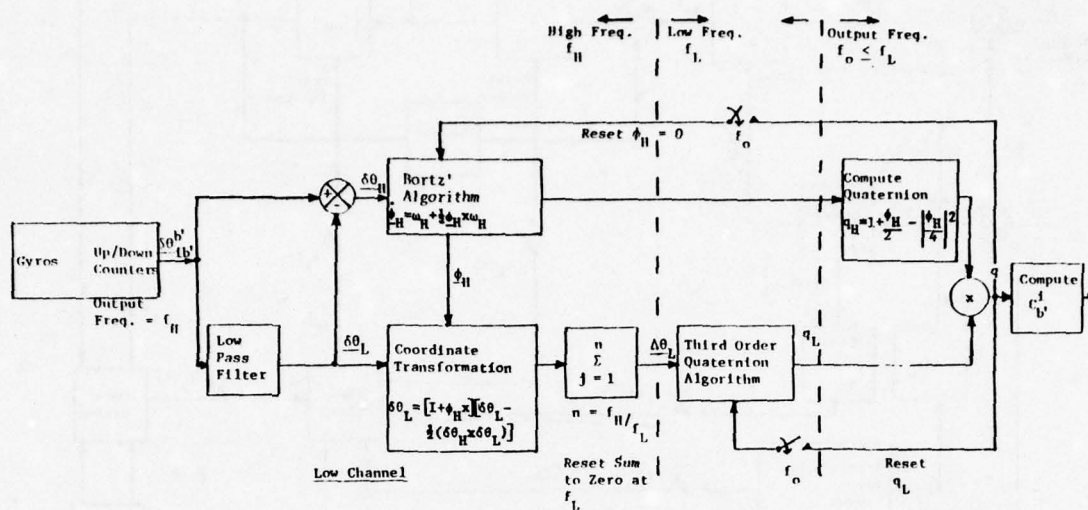


FIGURE 5 DUAL CHANNEL ATTITUDE ALGORITHM MECHANIZATION

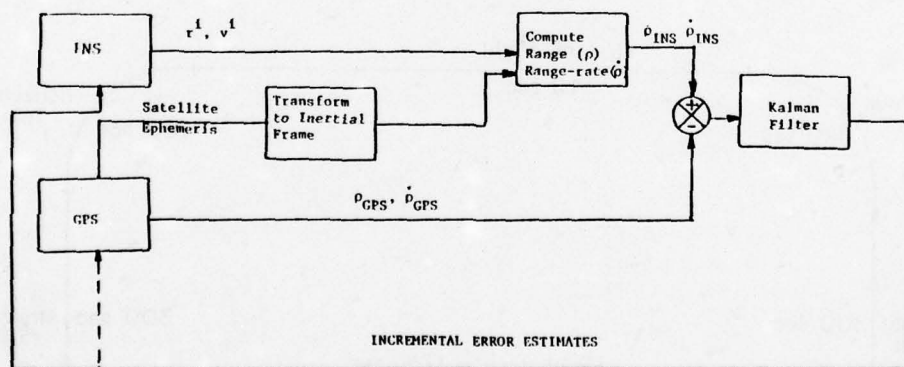


FIGURE 6 KALMAN FILTER CONFIGURATION FOR THE GPS AIDED INS [1]

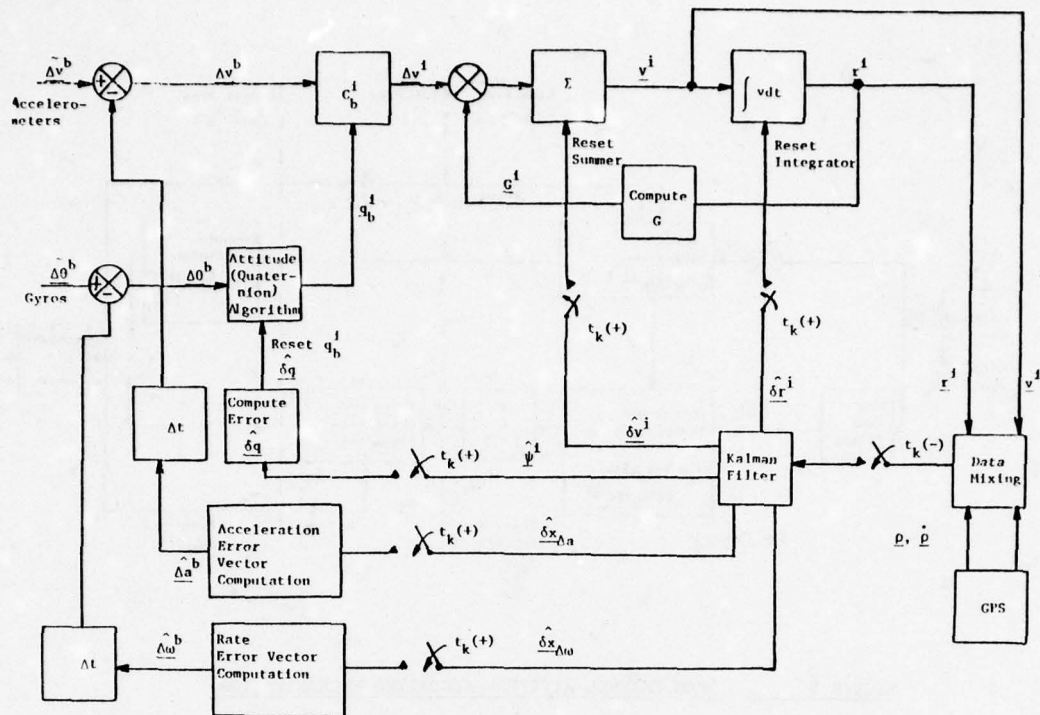


FIGURE 7 GPS/INERTIAL INTEGRATION SHOWING
ERROR ESTIMATE FEEDBACK [1]
(Baro/Inertial Loop not Shown)

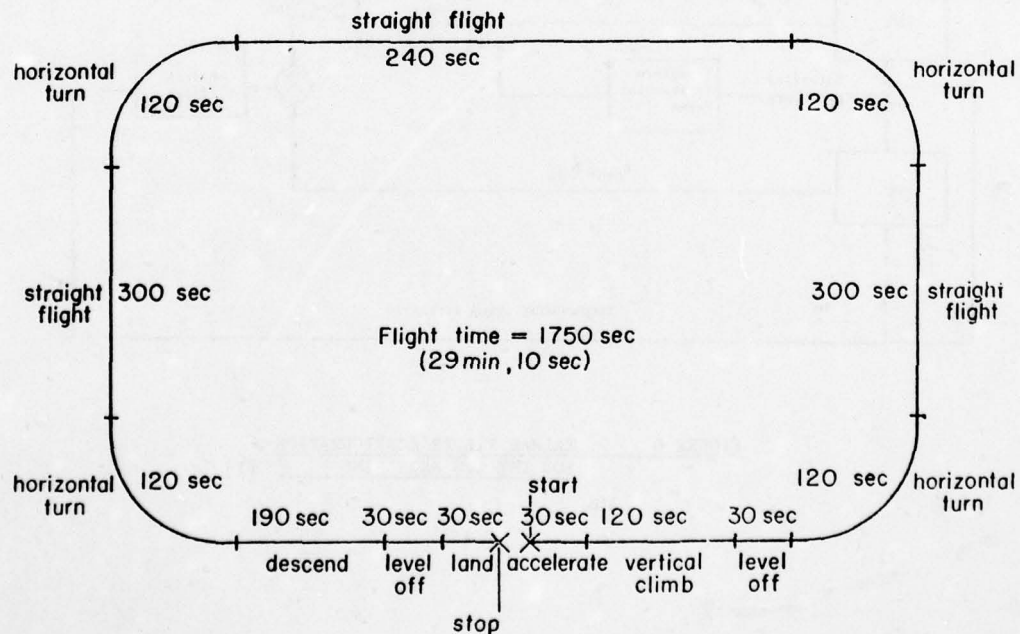


FIGURE 8 "RACETRACK" FLIGHT PROFILE

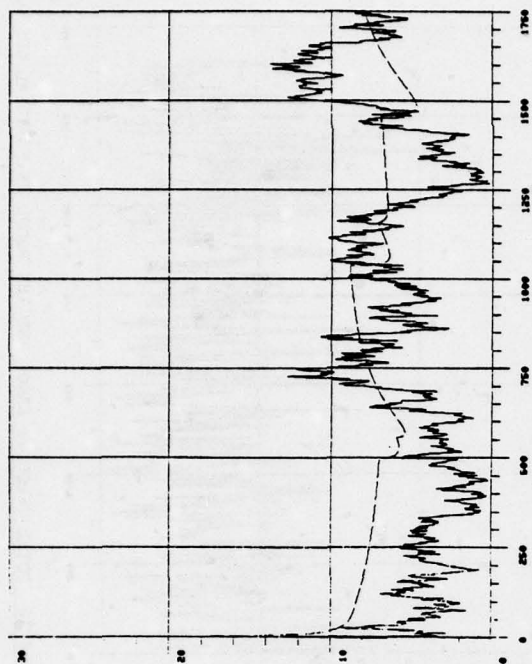


FIGURE 11 TOTAL POSITION ERROR AND RMS BOUND FOR 49 STATE FILTER

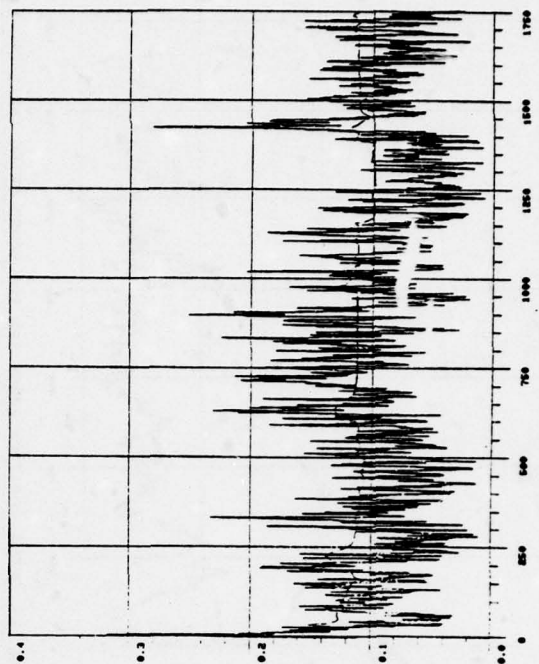


FIGURE 12 TOTAL VELOCITY ERROR AND RMS BOUND FOR 49 STATE FILTER

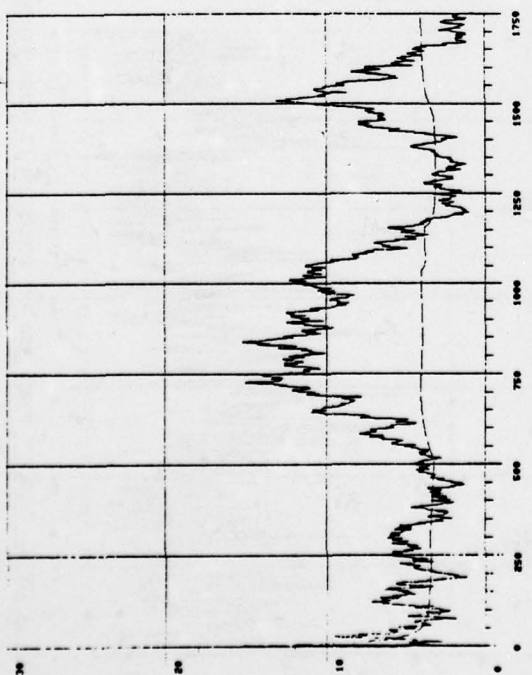


FIGURE 9 - TOTAL POSITION ERROR AND RMS BOUND FOR 45 STATE FILTER

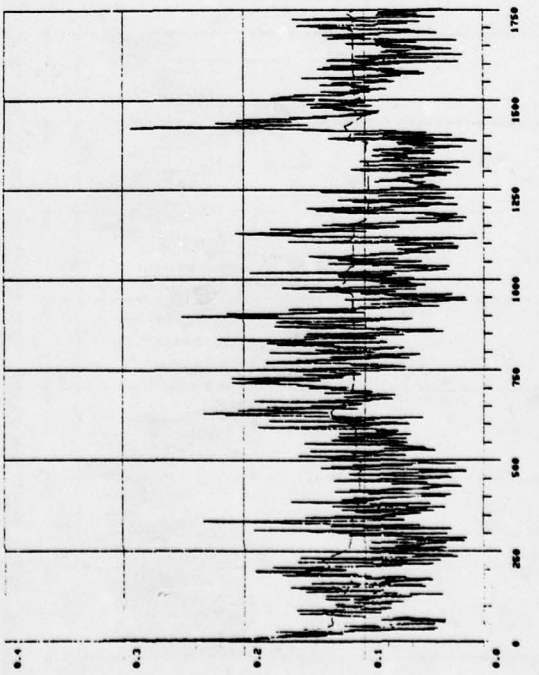


FIGURE 10 - TOTAL VELOCITY ERROR AND RMS BOUND FOR 45 STATE FILTER

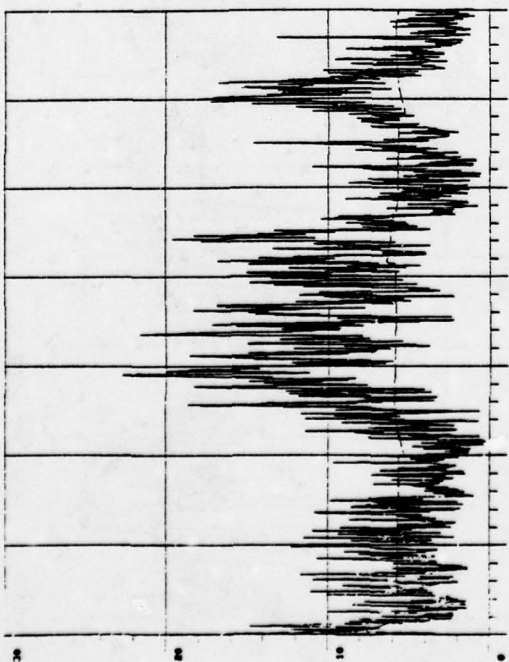


FIGURE 13 TOTAL POSITION ERROR AND RMS BOUND FOR 47 STATE FILTER

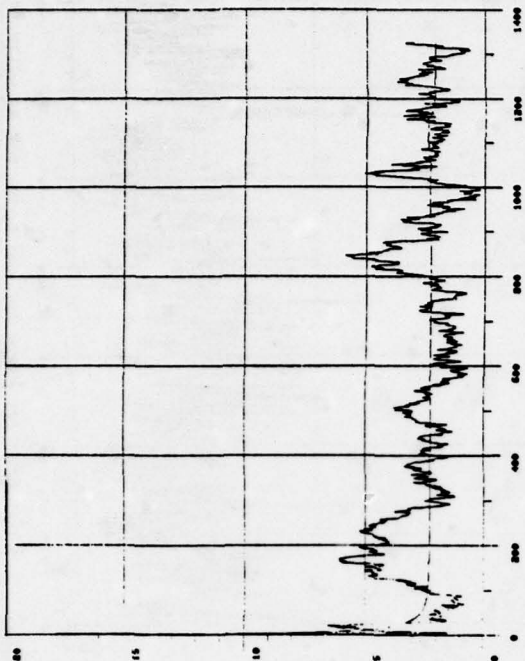


FIGURE 15 TOTAL POSITION ERROR AND RMS BOUND USING IDEAL GPS DATA

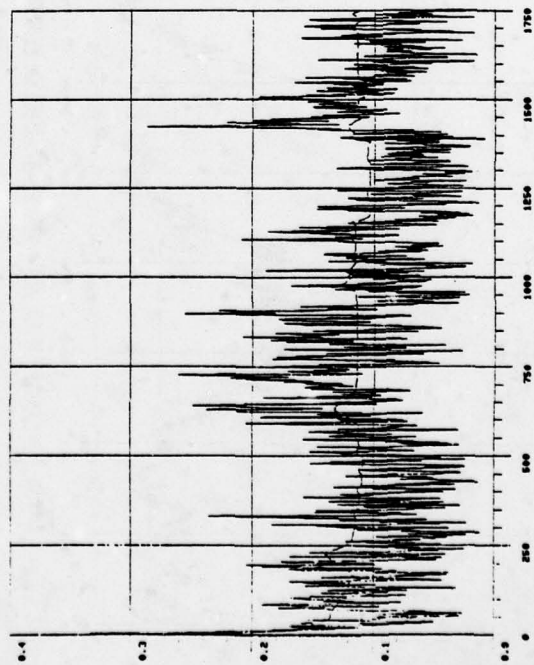


FIGURE 14 TOTAL VELOCITY ERROR AND RMS BOUND FOR 47 STATE FILTER

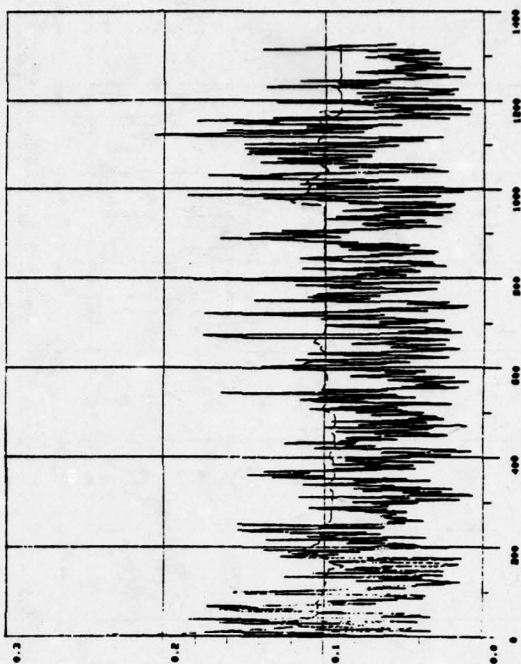


FIGURE 16 TOTAL VELOCITY ERROR AND RMS BOUND USING IDEAL GPS DATA

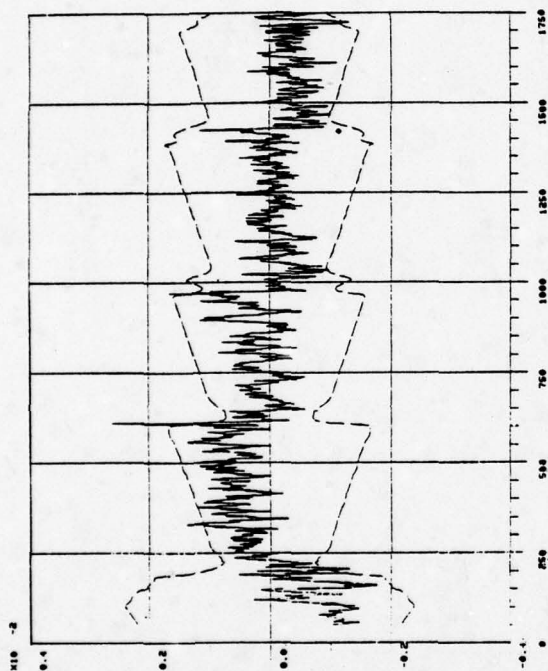


FIGURE 17 INERTIAL X-AXIS MISALIGNMENT ERROR AND RMS BOUNDS FOR 45 STATE FILTER

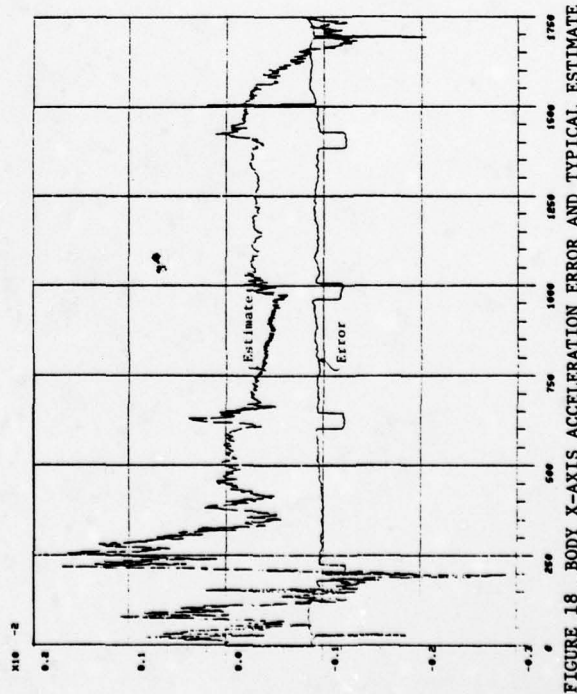


FIGURE 18 BODY X-AXIS ACCELERATION ERROR AND TYPICAL ESTIMATE

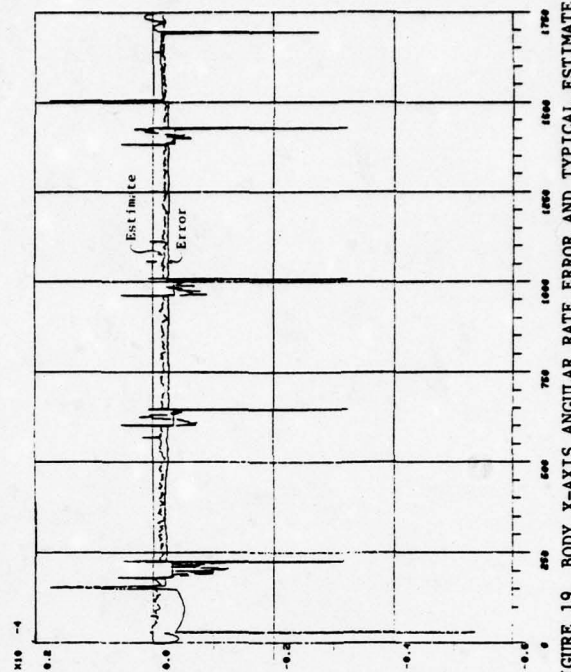


FIGURE 19 BODY X-AXIS ANGULAR RATE ERROR AND TYPICAL ESTIMATE

DIGITAL ARRAY SIGNAL PROCESSING TECHNIQUES APPLIED TO GUIDANCE AND NAVIGATION

by S. BLOCH

STANDARD ELEKTRIK LORENZ AG (ITT)
D-7000 STUTTGART 40, GERMANY

SUMMARY

A great many systems make use of highly versatile digital signal processing methods. In most cases filtering operations are performed in which the signal spectrum is altered. Pulse compression, MTI clutter cancelling, FFT spectral analysis etc. are typical examples for such applications.

In this paper spatial filtering, in particular virtual beam forming techniques, appropriate for guidance and precision landing operations, will be discussed.

Systems operating at L-Band will be briefly described demonstrating that when appropriate digital signal processing is applied, accuracy comparable to that of corresponding C-Band systems is achievable. Thus, the advantages of relatively low L-Band frequencies, may be exploited while maintaining a high degree of precision.

1. INTRODUCTION

The performance of Guidance and Microwave Landing Systems is determined by their capability to provide high angular resolution in azimuth and elevation.

This resolution is required in order to mitigate the effects of multipath interference caused by specular signal reflections.

In contrast to non-coherent noise and jamming, multipath reflections are phase coherent with the signals being evaluated. Therefore, they cannot be rejected by applying spectral analysis and matched filtering. As filtering in the frequency domain becomes inappropriate, temporal and spatial filtering methods gain significance. For this reason leading edge processors, data trackers and antenna arrays need to be incorporated in order to enhance the performance of high precision systems.

2. SPATIAL FILTERING

Spatial filtering reduces the amount of interference picked up by the receiving unit. Adjustable patterns are necessary in order to match the antenna characteristic to the signal/interference conditions. This operation requires in general a good angular resolution. Thus, antenna apertures much larger than the wavelength need to be utilized. High resolution is readily achieved at high microwave frequencies (e.g. C-Band and X-Band). However, the implementation becomes difficult when operation at lower frequencies is required. In such cases excessively large and costly antenna installations are needed. Special beam forming techniques, dubbed Digital Array Signal Processing (DASP), are instrumental in reducing the size and complexity of such antennas. They basically consist of mathematical algorithms which, in conjunction with antenna arrays, permit the generation of virtual (non physical) antenna patterns. DASP techniques exploit the high degree of flexibility and sophistication inherent to digital processors and thus make possible a considerable reduction in hardware cost.

The adaptability provided by DASP enables an operation which is matched to the variable multipath conditions along the flight path.

3. DIGITAL ARRAY SIGNAL PROCESSING (DASP)

Figure 1 depicts a conventional linear phased array consisting of N antenna elements. Depending on the angle of incidence θ , the wavelength λ and the array spacing D the signal phase at a given element is:

$$\varphi_n = \varphi_0 + \frac{2\pi D n}{\lambda} \sin\theta \quad n = 0, 1, 2, 3 \dots N-1$$

(φ_0 is the signal phase at element #0)

(φ_n is the signal phase at element #n)

Narrow band signals received by the array can be denoted by the following complex vector:

$$\underline{U} = \begin{pmatrix} U_0(t) \\ U_1(t) \\ \vdots \\ U_{N-1}(t) \end{pmatrix} = \underline{a}(t) \begin{pmatrix} 1 \\ \exp \frac{j2\pi D}{\lambda} \sin\theta \\ \vdots \\ \exp \frac{j2\pi(N-1)D}{\lambda} \sin\theta \end{pmatrix}$$

Beam forming is performed by introducing appropriate phase shifting as well as amplitude weighting in each of the N array channels.

The resulting signals are superimposed, thus yielding:

$$x(t) = \sum_{n=0}^{N-1} \beta_n U_n(t) e^{-j\alpha_n}$$

I.e. signal $x(t)$ is the output of the directional antenna. The antenna pattern is adjustable in shape and

steering angle, this being accomplished by an appropriate setting of the parameters β_n (amplitude weighting) and α_n (phase shift). A suitable choice of these parameters makes possible an optimization of the signal/interference ratio at the receiver input. Phased arrays require the utilization of analogue RF-components (e.g. variable attenuators and phase shifters). These components need to be carefully aligned and continuously monitored. The flexibility of phased arrays, though superior to that of mechanically steered antennas, is still rather restricted. A high degree of versatility may be achieved by using a DASP system as shown in figure 2. Here each element of the linear array is provided with an individual coherent receiver. The N array signals are simultaneously mixed down and A/D-converted at I.F.-level. In order to retain the full amplitude and phase distribution across the aperture, coherent reception is performed. Each receiver derives the inphase and quadrature components of the signal at its input. Thus, the full information of the incident field is recorded and digitally stored in the form of the afore-mentioned complex vector \underline{U} . This vector presents a hologram of the RF field incident upon the antenna. All further processing operations are performed using a fast digital computer. It is obvious that beam forming is readily implemented, as only straight forward arithmetic is involved in the computation of the variable:

$$x = \sum_{n=0}^{N-1} \beta_n U_n(t_0) e^{-i\alpha_n}$$

The computer is opted to choose between a variety of different actions.

A single pulse of unknown incidence (and thus a single set of data \underline{U}) can, for example, be repeatedly evaluated while iterating the parameters β_n and α_n . This procedure is appropriate in optimizing the signal/interference ratio. Such an adaptable monopulse operation cannot be provided by conventional phased arrays.

Another significant advantage of DASP systems is their insensitivity to misalignments of the hardware installations. Any erroneous deviation in phase or amplitude can be easily compensated for by simply correcting the digitally stored signal vector \underline{U} .

The following examples should give an impression of the efficiency and flexibility of DASP techniques.

Fourier Analysis

The mobile L-Band system shown in figure 3 is tailored to the area and en-route navigational needs of military aircraft. Its operational performance and accuracy are much superior to those of the currently used TACAN and VOR/DME systems.

Improved immunity against multipath signals is achieved by using a large antenna baseline. The diameter of the circular array shown is in the order of 6 wavelengths.

Addressed aircraft interrogations are received simultaneously by the N elements of the array. The signals are coherently demodulated and A/D-converted. A fast digital processor then computes the angle of incidence of the received RF-signals. The addressed ground station eventually retransmits the evaluated bearing information to the interrogating aircraft. In order to confine the ground response to the direction of the airborne station a configuration of 8 transmitting antennas, each covering an angular sector of 60° , is used. The time delay between signal reception and data retransmission is precisely defined. Thus, straight forward elapsed time ranging may be performed by the airborne unit.

A single interrogation is sufficient in providing the aircraft with the p/θ-information needed for a position fix. In contrast to TACAN the ground station transmits only upon request. Airborne interrogators and a large number of ground stations may therefore operate simultaneously on the same frequency channel.

The afore-mentioned determination of the angle of incidence requires the computation of the signal phase at each of the N array elements. The digitally recorded signal information at the outputs of the receiver multiple is evaluated:

$$\varphi_n = \tan^{-1} \left(\frac{U_{Qn}}{U_{In}} \right) \quad n = 0, 1, 2 \dots N-1$$

U_{Qn} and U_{In} are the inphase and quadrature components of the signal received by the n'th array element. An appropriate ROM table is used in order to expedite the computation of the trigonometric function. A phase behavior, as shown in figure 4a, is typical for the "clean environment" operation. The signal phases recorded at the outputs of the N receivers present the 2π -Modulo value of the actual phases. In order to take advantage of the large antenna baseline a reconstruction of the unambiguous phase needs to be introduced (fig. 4b). This operation requires the monitoring of phase transients between adjacent channels. Irregular transitions indicate the need for a 360° correction of the measured phase. In contrast to a corresponding analogue system this operation is easily implemented.

As long as the reception remains unperturbed by noise or multipath, the unambiguous phase behaviour is given by the following cosine-relationship:

$$\varphi_n = \frac{180}{\lambda} D \cos(\gamma - \frac{360}{N} n) \quad n = 0, 1, 2 \dots N-1$$

(This relationship is true for low elevations.)

γ = Bearing angle of the interrogating aircraft relative to an axis defined by the center of the circular array and antenna element #0.

D = Diameter of the array.

λ = Wavelength (0,3 m at L-Band).

A Fourier analysis performed on the data set $\underline{\Phi} = (\varphi_0, \varphi_1, \varphi_2 \dots \varphi_{N-1})^T$ permits the determination of the bearing angle:

$$\gamma = \tan^{-1} \frac{\sum_{n=0}^{N-1} \varphi_n \sin \left(\frac{360}{N} n \right)}{\sum_{n=0}^{N-1} \varphi_n \cos \left(\frac{360}{N} n \right)}$$

This procedure requires the computation of the real and imaginary parts of the first order Fourier coefficient.

Higher order Fourier coefficients provide very useful monitoring information. Any excessive perturbation of the phase vector ϕ , caused either by external sources (noise, multipath, jamming) or by system failure, gives rise to harmonics. The magnitude and distribution of the respective Fourier coefficients are appropriate in monitoring the status of the system. In cases where NOGO is indicated it is often possible to recognize the cause of the reduced performance.

Fast Fourier Transform algorithms are very efficient in performing the Harmonic Analysis described, and provide a significant reduction in the computational load.

The introduction of a digital signal processor enables the system to continuously perform self calibration. An antenna element in the center of the circular array couples special test signals into the N receiving channels. These signals are used to update a correction table which is stored by the computer. Any phase and amplitude deviation is being arithmetically compensated for. This approach deletes the need for precise alignment of the system.

DASP Interferometer

Another system demonstrating the versatility of DASP methods is shown in figure 5. The depicted linear array and its respective transmitter unit are parts of an air derived Microwave Landing System, which has been developed for the German Airforce. The mobile ground installation of this MLS generates signals which, when being processed on board, provide the airborne units with precision 3D-information. A unique feature of this system is that beam forming operations are being accomplished by the airborne processor, although the coherent array antenna is located on the ground. An airborne digital computer performs all of the arithmetic operations related to the generation of virtual antenna patterns. These patterns are matched in shape and steering angle to the variable multipath conditions along the landing profile. The system makes extensive use of the already existing airborne TACAN set. On board only an appropriate digital processor is required in order to achieve the extended MLS capability. Figure 6 shows a combination of a TACAN receiver and an associated digital processing unit. Some details of the MLS computer are given in table 1. The MLS system is quite complex and therefore cannot be comprehensively discussed in a short paper. For this reason only a brief description of the methods used in conveying precision elevation data will be given.

The vertical linear array shown in fig. 5 uses 40 identical low gain array elements. An additional element (located at the bottom of the mast) serves as a reference antenna. The overall length of the mast is 6 m, corresponding to an antenna aperture of approximately 24 wavelengths. The ground station (elevation) generates pulse trains consisting of 40 discrete RF bursts. During each burst two L-Band carriers are simultaneously transmitted with their frequencies differing by 100 KHz. Upon reception the two signals interfere, thus, giving rise to periodical 100 kHz fading.

One of the two signals mentioned establishes a phase reference. It is thus permanently being transmitted by the reference antenna. The second signal is being switched, within each single burst, from the reference antenna to one of the 40 array elements. The resulting change in the fading behaviour corresponds to the relative RF-phase:

$$\varphi_n = \frac{2\pi D n}{\lambda} \sin \theta$$

D = spacing between adjacent elements

= wavelength

θ = elevation angle

n = the number of the respective array element

With each new burst a different array element is selected. Thus, the transmission of 40 RF pulses enables a sequential information transfer of the amplitude and phase distribution across the array. The received data is A/D converted and recorded by the airborne digital processor in the form of a complex vector:

$$\underline{U} = \begin{pmatrix} U_1 \\ U_2 \\ \vdots \\ U_{40} \end{pmatrix} = \begin{pmatrix} A_1 e^{i\varphi_1} \\ A_2 e^{i\varphi_2} \\ \vdots \\ A_{40} e^{i\varphi_{40}} \end{pmatrix}$$

Figure 7 shows a phase and amplitude behavior which characterizes the (ideal) undisturbed reception. Here the phase of the individual array signals grows linearly from one element to another while the amplitudes remain at the same magnitude.

Apart from a simple procedure which helps to resolve the 2π phase ambiguities, no further measures need be taken to evaluate the aircraft's elevation.

$$\theta = \sin^{-1} \left(\frac{\lambda \varphi_{40}}{80 \pi D} \right)$$

Unfortunately the signal reception is always degraded by coherent ground reflections. The resulting distortion of the phase and amplitude distribution is shown in figure 8. In such cases a simple interferometric evaluation becomes exceedingly inaccurate.

A significantly improved operation is achieved by introducing the following beam forming method: The complex vector \underline{u} is divided into two subgroups. The following mathematical operations are then carried out:

$$x_1 = \sum_{n=1}^{20} U_n \beta_n e^{-i\alpha n} \quad x_2 = \sum_{n=1}^{20} U_{20+n} \beta_n e^{-i\alpha n}$$

β_n is a symmetrical sequence of 20 tapering coefficients. Similar to conventional antennas the introduction of these coefficients is appropriate in reducing the sidelobe problem. The complex variables x_1 and x_2 are equivalent to received signals of a ground station using two transmitting fan beam antennas. x_1 and x_2 , thus denote two virtual beams which are directional in the vertical plane. The computed "antenna patterns" may be tilted in elevation simply by altering the parameter α . They are vertically steered so as to minimize the effects of ground interferences. In this process the coherent phase at the center of each array subgroup is retained, permitting interferometric determination of the aspect angle θ . The baseline of the resulting interferometer corresponds to the 20 D spacing between the two antenna groups.

$$\begin{aligned} \varphi_1 &= \tan^{-1} \left[\frac{\text{Im}(x_1)}{\text{Re}(x_1)} \right] \\ \varphi_2 &= \tan^{-1} \left[\frac{\text{Im}(x_2)}{\text{Re}(x_2)} \right] \\ \theta &= \sin^{-1} \left(\frac{\lambda(\varphi_2 - \varphi_1)}{40 \pi D} \right) \end{aligned}$$

A problem typical to all large baseline interferometers is that their information is highly ambiguous. This problem necessitates the introduction of a special acquisition procedure. Here once again digital signal processing techniques are very useful, as they enable simultaneous computation of various interferometers of different sizes. Thus, ambiguities are readily resolved.

Ambiguity resolution is needed only during the initial phase when the airborne set commences communicating with the ground station. Once a correct elevation has been established, an appropriate data tracking circuit is activated making possible continuous ambiguity monitoring. (The tracker mentioned presents one of the temporal filtering measures which further enhances the system's accuracy. Temporal filtering is not treated in this paper it is, however, worth mentioning that this function is also performed by the on-board digital computer.)

Multiple Stacked Beam System

Fast Fourier Transform algorithms provide a very powerful computational tool. They enable an implementation of digital signal processing methods which may be referred to as virtual stacked beam techniques. This approach requires the computation of a large number of virtual beams. Thus:

$$\begin{aligned} x_1 &= \sum_{n=1}^N U_n e^{-\frac{i2\pi n}{N}} \\ x_2 &= \sum_{n=1}^N U_n e^{-\frac{i4\pi n}{N}} \\ x_3 &= \sum_{n=1}^N U_n e^{-\frac{i6\pi n}{N}} \\ &\vdots \\ x_K &= \sum_{n=1}^N U_n e^{-\frac{i2\pi K n}{N}} \\ &\vdots \\ x_N &= \sum_{n=1}^N U_n e^{-\frac{i2\pi N n}{N}} = \sum_{n=1}^N U_n \end{aligned}$$

The set of complex variables \underline{x} represents a system of N interlacing virtual antenna patterns (as shown in figure 9).

In contrast to the afore-mentioned interferometer approach the array aperture need not be partitioned. Each individual antenna beam is a product of the entire array aperture. In comparison to the interferometer approach an improved angular resolution can be attained. It appears at first glance as if N^2 complex multiplications (each consisting of 4 real multiplications and 2 summations) are required in order to compute N overlapping beams. This arithmetical load may, however, be significantly reduced when appropriate FFT-algorithms are used. If for example 32 of the array signals are evaluated, the number of complex multiplications which are required to compute 32 antenna beams may be reduced from the original 1024 to 80.

The variables x_1 through x_{16} correspond to virtual beams monitoring the positive elevational space. Whenever a signal is received, the two adjacent beams with the largest signal magnitude are determined. An appropriate interpolation between these two beams is then performed. This interpolation, which is similar to the procedure used by monopulse trackers, provides a precise determination of the elevation angle θ . The beams numbered 17 through 31 correspond to negative elevations. These beams provide useful information about the specular ground reflections. Using this information the entire multiple beam pattern

may be tilted so as to establish an antenna null at the negative angle from which the ground multipath is incident.

The multiple stacked beam method is completely unambiguous and, thus, requires no extra clearance procedure.

Lateral Diversity + Null Steering

The performance of Microwave Landing Systems (MLS) may be further improved by deploying two dimensional antenna patterns. Virtual pencil beams rather than fan beams are used, providing additional discrimination against lateral interference. This measure is especially useful when operating at very low elevations. In such cases in-beam signals, reflected by large buildings and hangars, present a serious problem to the MLS operation. They give rise to lateral reflections which are incident from positive elevations. Unlike ground reflections this type of coherent interference cannot be discriminated against by fan beams. It is therefore also required to extend the angular resolution to the horizontal plane. Pencil beam antennas necessitate the utilization of planar arrays which are, in general, more complex than the aforementioned linear array configuration. For landing operations, however, thinned arrays may be used. The large sidelobes caused by the reduction in the number of array elements are not overly troublesome providing they can be confined to angular regions where no specular reflections are expected. Figure 10 shows a configuration of 31 array elements distributed across a planar aperture of $4.5 \times 25\lambda$. The pencil beam generated by this antenna is shown in figure 11. Considering the very small number of array elements used, this configuration may be regarded as optimum. Here, the angular space -8° to $+5^\circ$, which is especially sensitive to multipath interference, is cleared of large antenna sidelobes.

The coherent antenna array shown, is a part of a stationary ground derived Microwave Landing System proposed to ICAO by the German Government as a successor to the present ILS. The ground installations of this MLS receive DME interrogations of landing aircraft. The 3D information is processed by computers located in the ground station with the result eventually transferred, via a data link, to the interrogating airborne station.

As signal processing is performed on the ground, a high degree of sophistication is attainable. Virtual pencil beams, monopulse interferometers and multiple stacked beams etc. are readily implemented. The two dimensional antenna pattern is directional both in elevation and azimuth. Null Steering schemes are introduced providing an improved immunity against ground reflections. This means, that a null in the virtual pattern is directed at the specific negative elevation from which reflected signals are incident. The system deploys an additional horizontal linear array which is used to provide coarse direction finding information. Once a signal is received and its approximate bearing established, the pencil beam is pointed at the aircraft. This operation is inhibited in directions where large reflecting structures are located. All steering positions that, due to intense reflections, must be avoided are stored in a special ROM-table. The implementation of such conditional target following schemes, permits multipath rejection tailored to the specific interference conditions of the given site.

Adaptability and Modularity

The crossed array configuration shown in figure 12 provides azimuth information and is designed to satisfy the most stringent MLS requirements in terms of accuracy and integrity permitting full Cat. III operation.

The antenna installation depicted consists of 29 low cost elements, each connected to an individual coherent receiver. The same antenna elements may be configured in different arrangements so as to match the complexity of the antenna installation to the operational requirements and the specific multipath conditions. Low cost systems differ from full capability versions in their geometrical arrangement and the number of elements used. System retrofits are relatively easy to implement as only straight forward alterations to the antenna array and the corresponding software need to be introduced. The modular structure of the system demonstrates convincingly the high flexibility which may be achieved when DASP techniques are deployed.

The two linear arrays of the antenna installation shown in fig. 13 are positioned longitudinally and orthogonally to the Runway Center Line (RCL). They perform three main operations:

- a) Provision of omnidirectional azimuth information comparable to Doppler-VOR (TMA-Navigation).
- b) Provision of precision azimuth within a sector symmetrical to the RCL (MLS guidance).
- c) Conversion of conical information into azimuth angle in a planar coordinate system.

The configuration shown has been designed to provide high angular accuracy when operating under very severe multipath conditions. The system is, therefore, appropriate for airport scenarios known to be exceptionally critical. Most airports will be less affected by lateral multipath. Their antenna installation may be simplified thus resulting in a considerable cost reduction. Fig. 13 shows an example of such a different antenna configuration. The arrangement shown at the top has the same baseline as that of the proposed high capability version (which is shown in the center section). It will therefore provide the same clean environment measurement results. The difference in hardware effort (17 receiver channels as compared to 29 channels) is directly related to hardware and computational costs. The reduced system can withstand interference in the order of 50 % M/D (Multipath to Direct Signal Ratio), whilst the full capability system remains effective even under severe conditions of 80 % M/D. For comparison, the angle ambiguity resolution of the TACAN system can theoretically withstand an M/D ratio of 31 %. (The indicated M/D-values present worst case estimates as they do not consider additional improvements which may be achieved by using leading edge processors and data trackers).

AD-A076 146

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/6 17/7
ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQU--ETC(U)
AUG 79

UNCLASSIFIED

AGARD-CP-272

NL

3 OF 4

ADA
076146

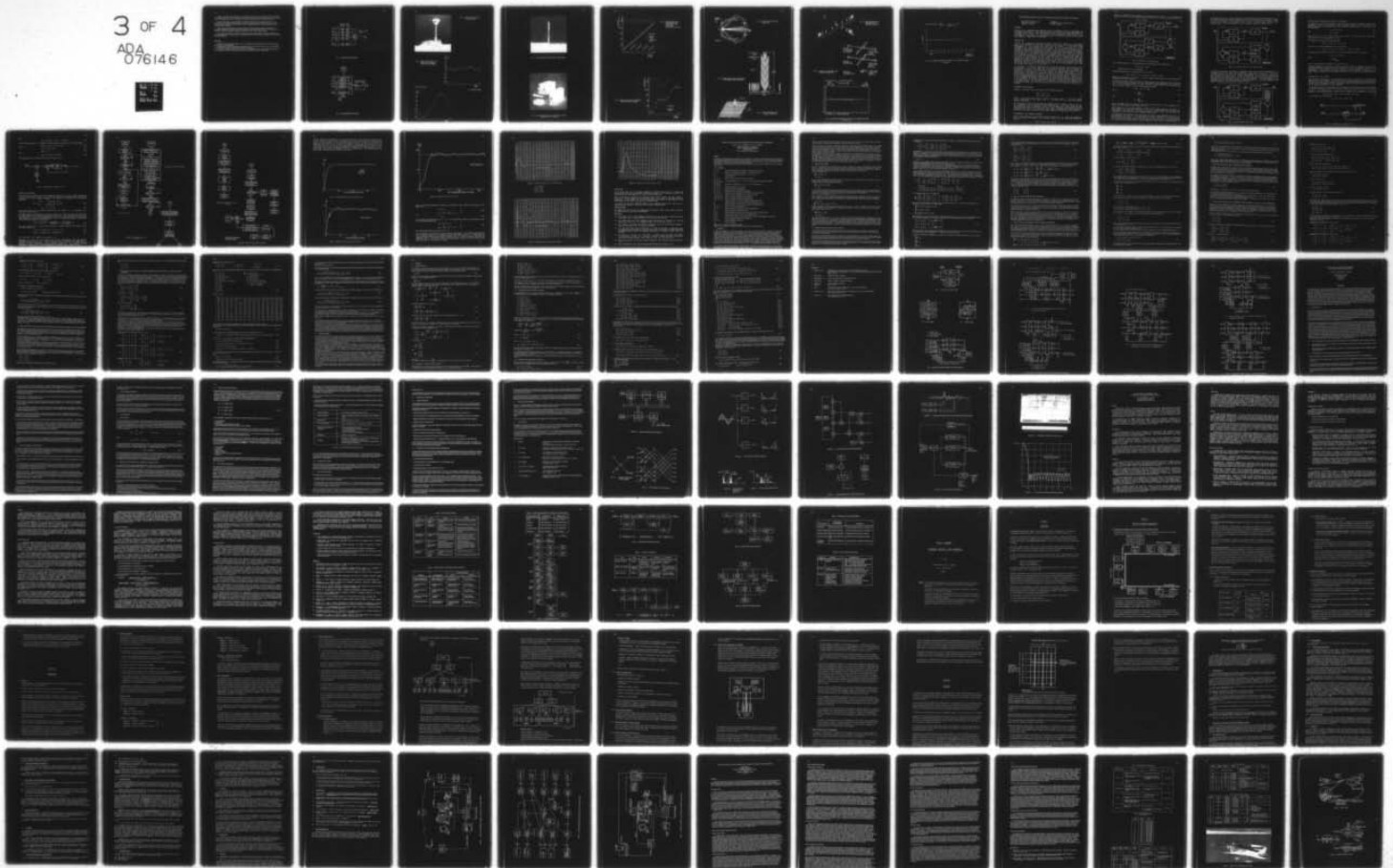


Figure 14 presents the behaviour of a L-Band MLS as plotted in a simulated approach and landing at Kennedy Airport, New York. The respective L-Band system incorporated a cruciform array at the stop-end of the runway and a vertical thinned planar array located abeam of the touch down point.

Similar simulations were conducted for different scenarios by the Technical University in Braunschweig. The complex multipath model used has been developed by the MIT-Lincoln Lab. in order to enable ICAO to compare the performance of three competing Microwave Landing Systems under the same multipath conditions.

Apart from the afore-mentioned L-Band system two C-Band candidate MLS were evaluated. The simulation demonstrated that the L-Band MLS, despite of the larger wave length of its signals, was capable in providing multipath rejection comparable to that of the C-Band systems.

The flexibility inherent to digital signal processing further permits an enhanced operation of the L-Band-MLS making it appropriate for additional tasks such as Direction Finding as well as TMA and enroute navigation.

CONCLUSION

Digital signal processing methods used in conjunction with coherent antenna array provide very powerful means of beam forming.

Systems with differing performance and complexity may be realized making use of the high degree of versatility and flexibility offered by digital processors.

The paper presented a number of different L-Band systems which by virtue of sophisticated arithmetic operations effectively mitigate the effects of coherent multipath interference.

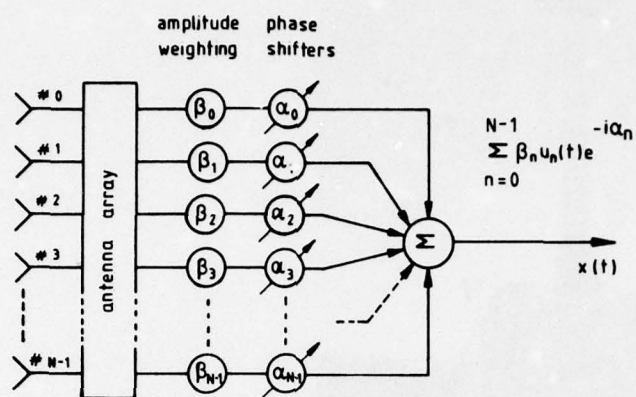


FIG. 1: CONVENTIONAL PHASED ARRAY

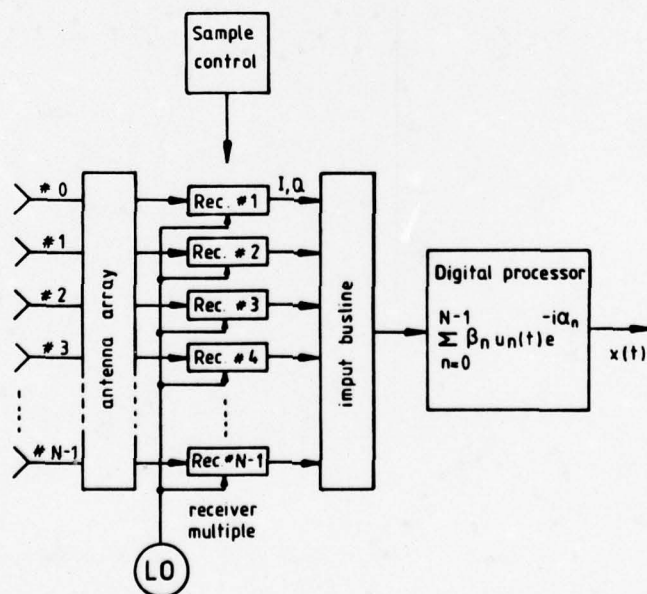
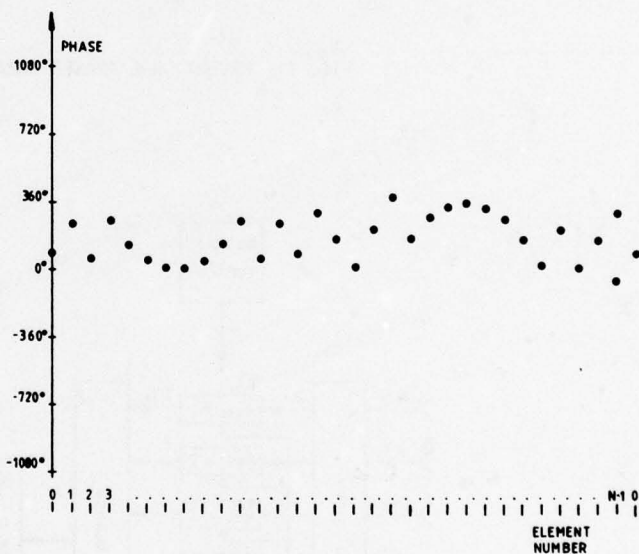


FIG. 2: DASP BEAM FORMING TECHNIQUE

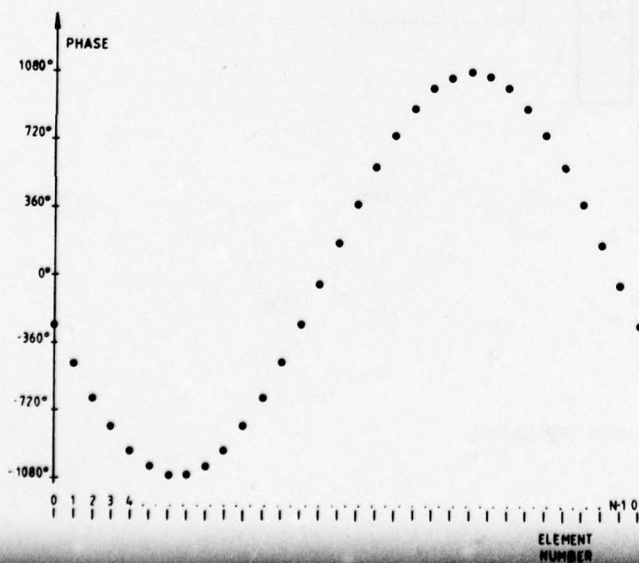


FIG. 3: DASP NAVIGATION SYSTEM
(CIRCULAR ARRAY)

FIG. 4: PHASE DISTRIBUTION AT THE
APERTURE OF A CIRCULAR
ARRAY (CLEAN ENVIROMENT)



b) resolved phase



a) unresolved phase

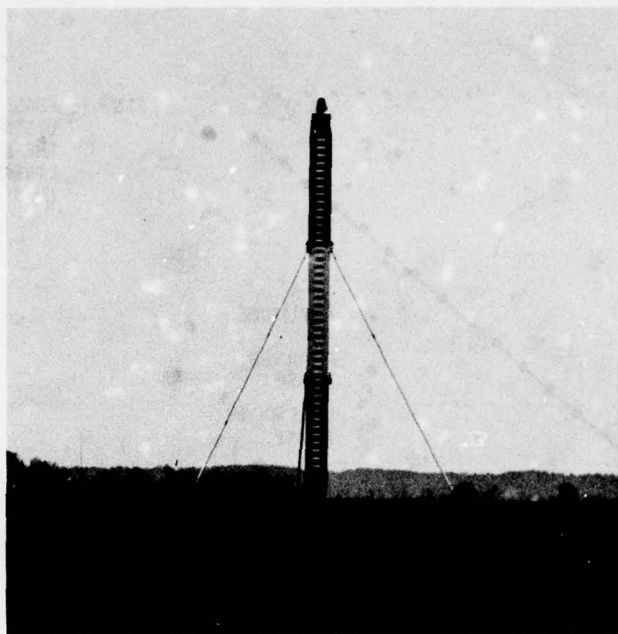


FIG. 5: DASP MICROWAVE LANDING SYSTEM (LINEAR ARRAY)

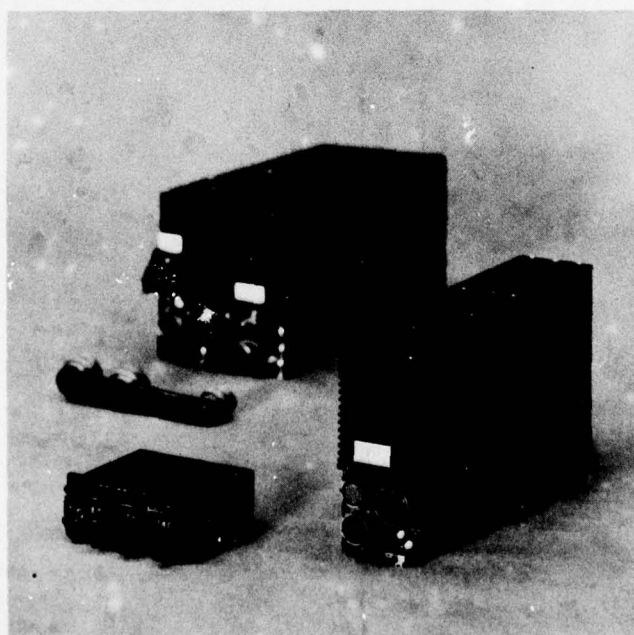


FIG. 6: TACAN RECIEVER WITH ITS ASSOCIATED MLS DIGITAL PROCESSING UNIT (foreground)

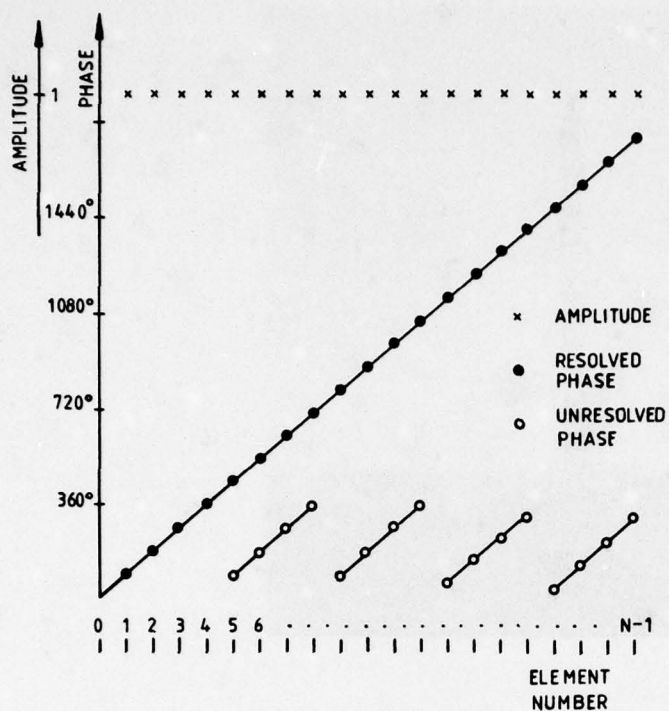


FIG. 7: PHASE AND AMPLITUDE DISTRIBUTION ACROSS THE APERTURE OF A LINEAR ARRAY. (Undisturbed conditions)

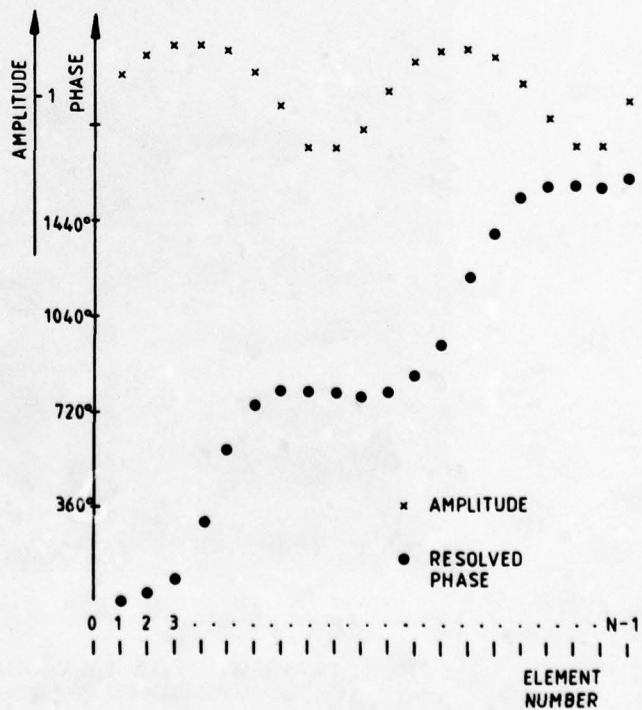


FIG. 8: PHASE AND AMPLITUDE DISTORTIONS CAUSED BY MULTIPATH SIGNALS

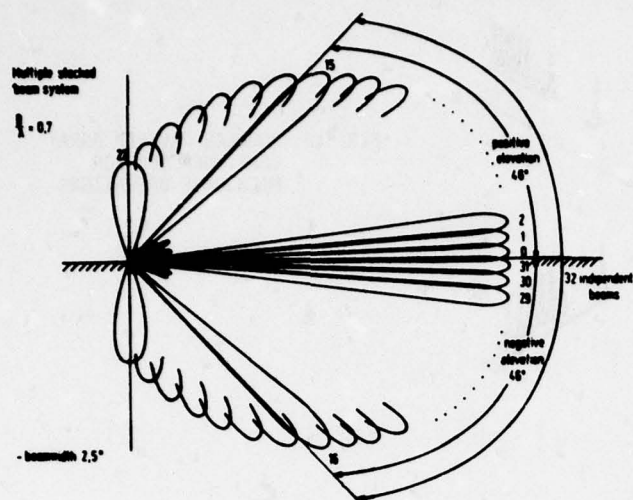


FIG. 9: VIRTUAL MULTIPLE STACKED BEAM SYSTEM

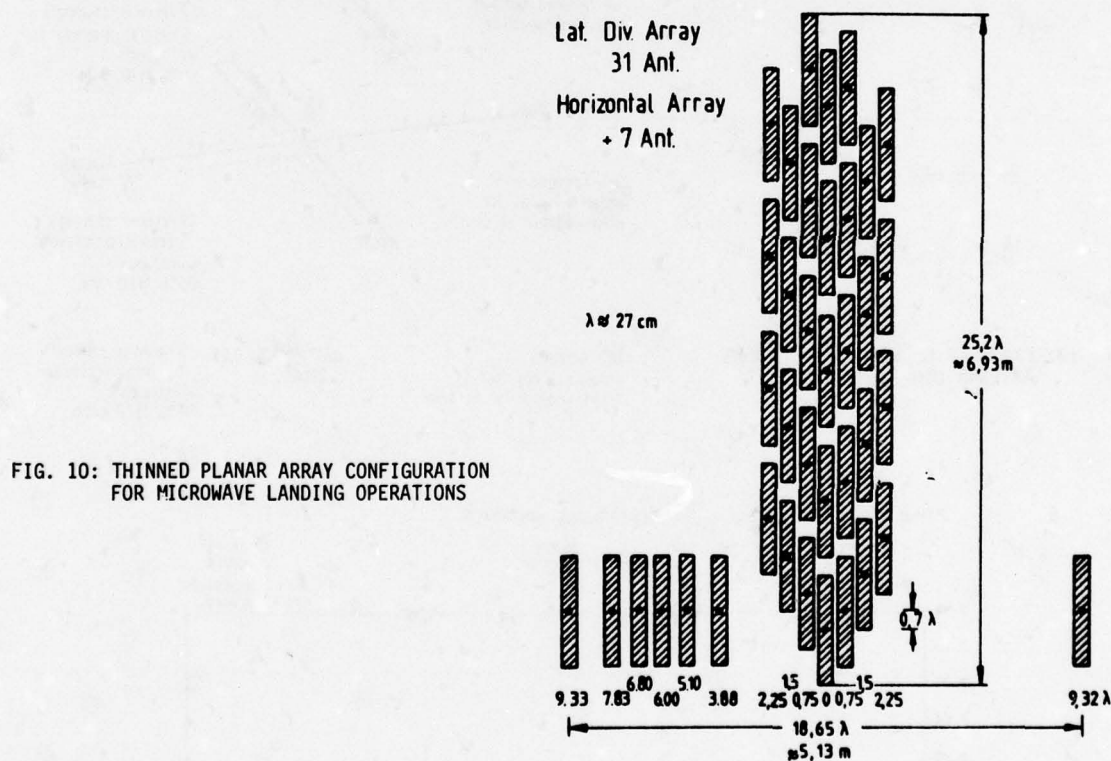


FIG. 10: THINNED PLANAR ARRAY CONFIGURATION FOR MICROWAVE LANDING OPERATIONS

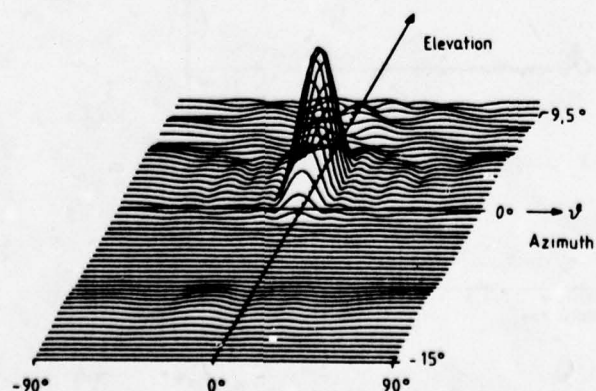


FIG. 11: PENCIL BEAM GENERATED BY A THINNED PLANAR ARRAY

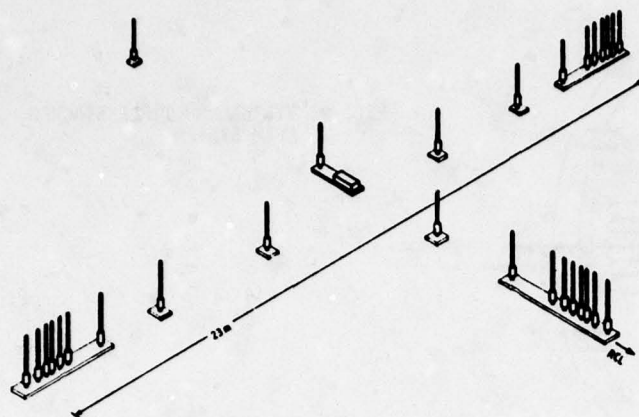


FIG. 12: MODULAR CROSSED ARRAY CONFIGURATION FOR MICROWAVE OPERATIONS

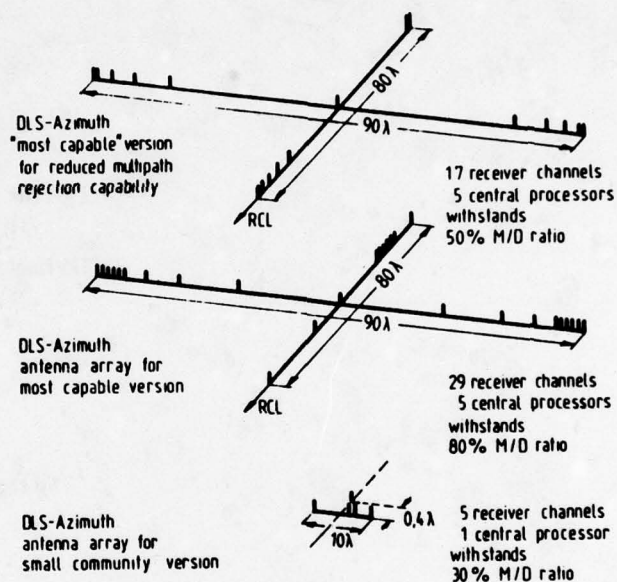
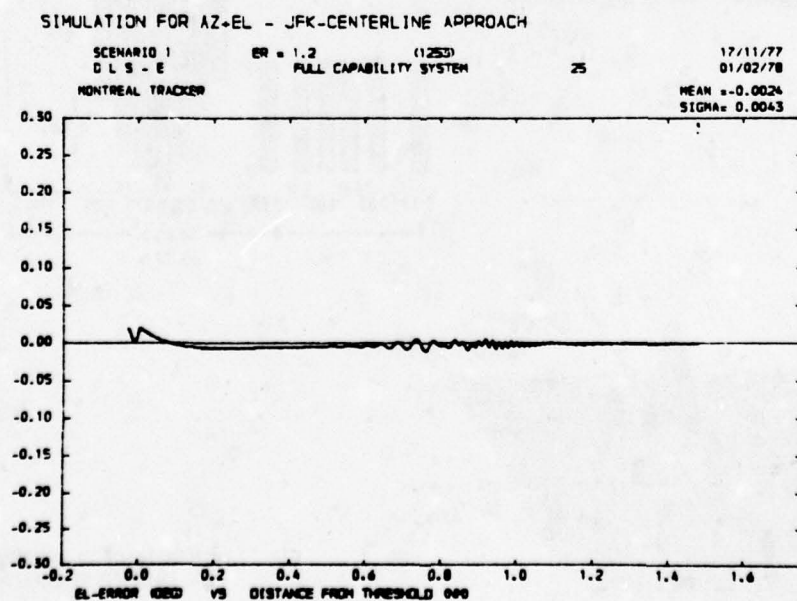


FIG. 13: EXAMPLES OF DIFFERENT LINEAR ANTENNA CONFIGURATIONS

FIG. 14a: SIMULATED APPROACH AND LANDING AT JFK-AIRPORT (NEW YORK)
(elevation)

MULTIPATH SIMULATION SCENARIO 1 ER = 13.0 C(1284)
DLS - A FULL CAPABILITY SYSTEM

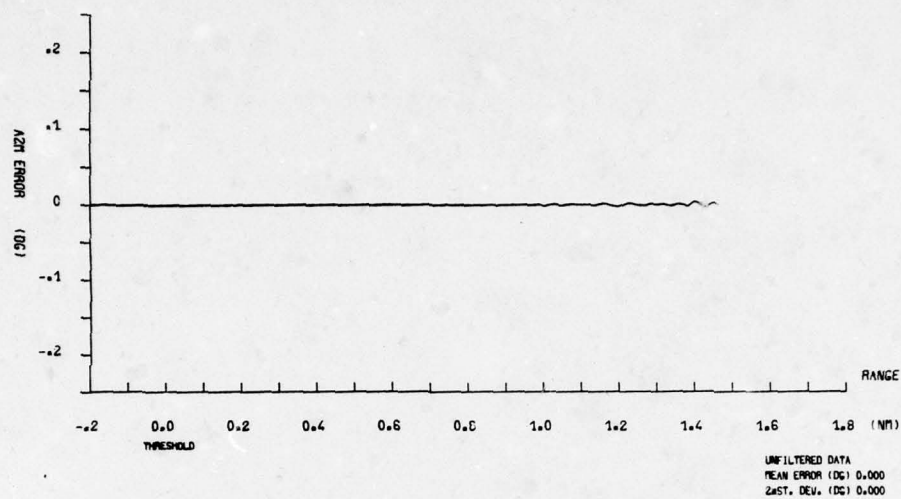


FIG. 14b: SIMULATED APPROACH AND LANDING AT JFK-AIRPORT (NEW YORK)
(azimuth)

MICROCOMPUTER-BASED ON-LINE STATE ESTIMATION WITH APPLICATIONS TO SATELLITES

N.K. Sinha and S.Y. Law
McMaster University
Hamilton, Canada

R. Mamen
Communications Research Centre
Ottawa, Canada

ABSTRACT

The use of a Luenberger-type observer with an adaptive filter is discussed for estimating the states of a nonlinear system from the noise-contaminated measurements of the output of the system. The proposed method requires much less computation than the extended Kalman filter and, therefore, can be implemented more easily on a microcomputer. Application to the determination of the orbital states of the unified state model of a satellite is considered.

INTRODUCTION

In order that corrective action may be taken to maintain a satellite in the desired orbit it is necessary to estimate accurately the orbital states from ground-based measurements of the elevation, the range and the azimuth. In practice, these measurements are often contaminated with noise, and in a recent paper [1] it was proposed to use a Kalman filter for this purpose. As the Kalman filter requires prior knowledge of the noise covariance matrix as well as the initial error covariance matrix, an adaptive approach for determining the optimum gain matrix was suggested, following earlier work [2]. The problem was further complicated by the fact that the unified state model [3] for the orbital trajectory is inherently nonlinear. This led to the use of the extended Kalman filter in which a linearized version of the nonlinear model was used with the stipulation that this linearized model is updated after each step utilizing the latest estimates of the states. As a result, the amount of computation required was rather excessive, and the accuracy of the estimate may be affected. Other methods of nonlinear state estimation (second-order filter and invariant embedding approaches) may give more accurate results, but require even more computation [4]. With the present state of art in the microcomputer area, none of these methods can be used for on-line state estimation in real time.

In this paper, an alternative approach to the problem of nonlinear state estimation will be presented. It is well known that the Luenberger observer can be utilized for determining the states of an observable linear system from noise-free measurements of the output. Furthermore, the order of this observer is generally lower than that of the Kalman filter, and therefore it further reduces the computational effort. If the measurements of the output are contaminated with noise, however, the Luenberger observer gives erroneous estimates of the state. One approach to overcome this difficulty is to filter the measurements before applying them to the Luenberger observer. Widrow et al. [5] have proposed an adaptive filter which may be used for this purpose, and in the case of linear systems, the results are comparable to those obtained using the Kalman filter. It is proposed to see how this method can be applied to the case of state estimation for a nonlinear system. In particular, it is desired to investigate the possibility of utilizing this method for on-line estimation of the states of a satellite using a microcomputer.

STATEMENT OF THE PROBLEM

Consider a nonlinear system described by the following equations

$$\dot{x}(t) = f(x, u, t) \quad (1)$$

$$y(t) = g(x) + v(t) \quad (2)$$

where $x \in R^n$ is the state vector, $u \in R^m$ is the control input, $y \in R^p$ is the output vector contaminated by the noise vector $v \in R^p$ and $f(\cdot)$, $g(\cdot)$ represent general nonlinear functions.

Our problem is to obtain the best possible estimate of $x(t)$, given $y(\tau)$ for $\tau \in [0, t]$. It is proposed to use an observer of the Luenberger type. It may be pointed out that the pioneering work of Luenberger deals with deterministic linear time-invariant continuous-time systems which are free from noise. Discrete-time observers will be considered here, since it is desired to use a digital computer for state estimation. Moreover, our system is nonlinear and the measurements are contaminated with noise.

DEVELOPMENT OF THE PROPOSED ALGORITHM

We shall first consider the noise-free case for which $v(t) = 0$. The block diagram in Fig. 1 illustrates an observer for a nonlinear system, with the following differential equation

$$\dot{\hat{x}}(t) = f(\hat{x}, u, t) + L(y - \hat{y}) \quad (3)$$

where $\hat{x}(t)$ represents the estimate of the state $x(t)$ at time t , L is a suitable gain matrix, and $\hat{y}(t)$ is the estimated output obtained from $\hat{x}(t)$ through the relationship

$$\hat{y}(t) = g(\hat{x}(t)) \quad (4)$$

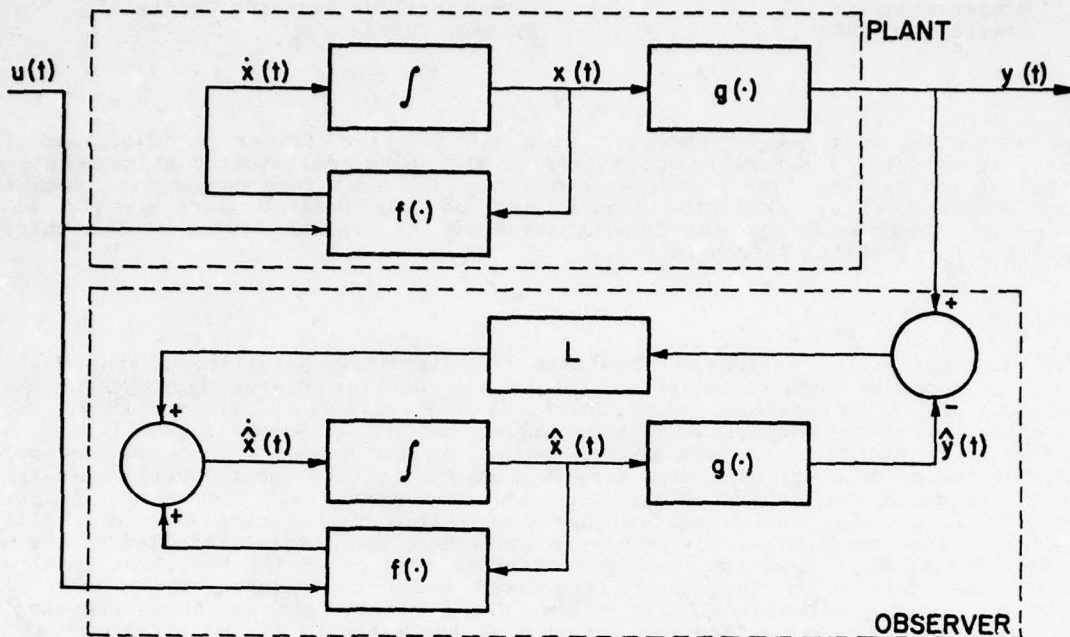


Fig. 1: Observer for a nonlinear system.

For a stable observer, it is desired that the estimation error, defined as

$$\tilde{x}(t) \triangleq x(t) - \hat{x}(t) \quad (5)$$

approach zero as t is increased.

Subtracting equation (3) from equation (1), we have

$$\dot{\tilde{x}}(t) = f(x, u, t) - f(\hat{x}, u, t) - Lg(x) + Lg(\hat{x}) \quad (6)$$

Hence, the problem is to select L in such a manner that equation (6) represents an asymptotically stable system.

It may be noted that unlike the case of linear systems, the right-hand side of equation (6) is not an explicit function of $\tilde{x}(t)$. The Taylor series expansion of $f(\cdot)$ and $g(\cdot)$ about a certain operating point, x_0 , may be used for simplification, and, retaining only the first-order terms, we get the following approximation

$$\dot{\tilde{x}}(t) = (A - LC) \tilde{x}(t) \quad (7)$$

where

$$A = \left. \frac{\partial f}{\partial x} \right|_{x=x_0} \quad (8)$$

and

$$C = \left. \frac{\partial g}{\partial x} \right|_{x=x_0} \quad (9)$$

Hence, one must select the matrix L such that the real parts of the eigenvalues of $(A-LC)$ have sufficiently large negative values.

Since equation (7) is an approximation, valid only for small changes in the state, it would appear that one must calculate A and C as the operating point changes, and recalculate L for each case. This would, therefore, require about the same amount of computation as is required for the extended Kalman filter.

It should be noted, however, that the actual location of the eigenvalues of $(A-LC)$ is not important, as long as it is understood that these eigenvalues are located to the left of a specified line $s = -a$ in the s -plane. One may, therefore, select the elements of the matrix L in such a manner that this condition for stability is satisfied even if the operating point varies within a large range of values. For very large changes in

the operating point, it may be necessary to adjust the values of the elements of L . In many practical problems it may be possible to divide the state-space into a small number of regions for each of which a given value of the L -matrix will be adequate. These values may be stored in the computer in the form of a look-up table, and used in an adaptive manner. The block diagram of such an adaptive scheme is shown in Fig. 2.

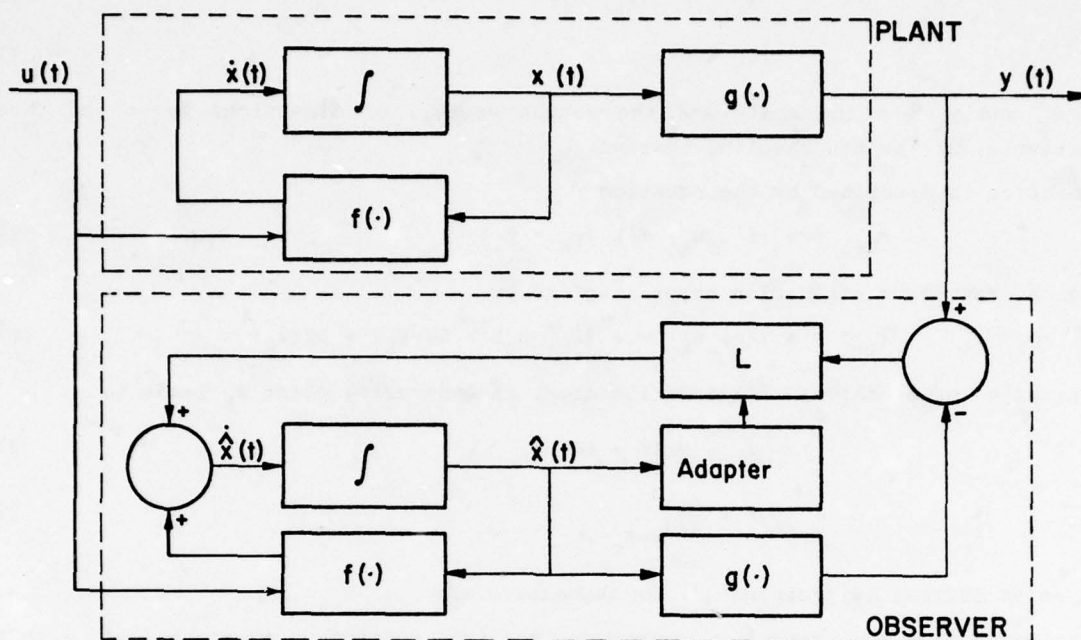


Fig. 2: An adaptive observer for a nonlinear system.

We shall now consider the effect of additive noise in the output measurements. If information about the statistical properties of the signal as well as the noise is available then one may filter the outputs before applying to the observer. This filter may be designed on the basis of the least mean-square error criterion, following the principles of the Wiener filter. It may be added that the requirement of knowledge of the statistics of the signal and the noise corresponds to the requirement of the knowledge of the covariance matrices in the Kalman filter. If such knowledge is not available a priori, one may use Widrow's adaptive filter [5]. The block diagram of the adaptive observer with the filter is shown in Fig. 3.

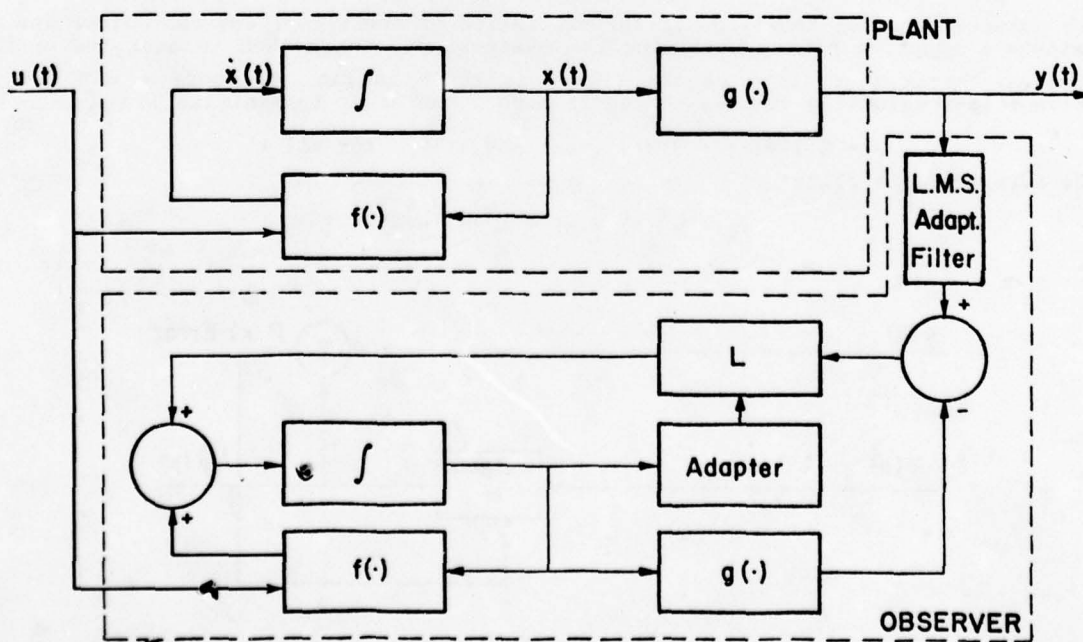


Fig. 3: Adaptive observer with filter.

APPLICATION TO ESTIMATING THE STATES OF A SATELLITE

The application of the above scheme to estimating the states of a satellite will now be considered. Since it is desired to use a digital computer, the first step is the discretization of the unified state model, represented by equations (1) and (2), to the following form

$$x_{k+1} = \phi(x_k, u_k) \quad (10)$$

and

$$y_k = g(x_k) \quad (11)$$

where x_k and y_k are the state and the output vector, of dimensions seven and three, respectively, at the k th sampling instant.

The observer is described by the equation

$$\hat{x}_{k+1} = \phi(\hat{x}_k, u_k) + L(y_k - \hat{y}_k) \quad (12)$$

and hence, the state estimation error is given by

$$\tilde{x}_{k+1} = \phi(x_k, u_k) - \phi(\hat{x}_k, u_k) - Lg(x_k) + Lg(\hat{x}_k) \quad (13)$$

Linearization of equations (10) and (13) about the operating point x_k leads to

$$\tilde{x}_{k+1} = (F - LC) \tilde{x}_k \quad (14)$$

where

$$F = \left. \frac{\partial \phi}{\partial x} \right|_{x=x_k} \quad (15)$$

and C is as defined by equation (9) but evaluated at x_k .

The implementation on a computer, therefore, requires setting up equation (12), with the observer weighting matrix L selected suitably to ensure stability by making the eigenvalues of the matrix $(F-LC)$ lie within a circle of specified radius $r < 1$. Furthermore, it is possible to store in the memory of the computer a look-up table so that L may be adjusted as large changes take place in the value of the estimated state \hat{x}_k .

THE ADAPTIVE L.M.S. FILTER

When the output measurements are contaminated with noise, these must be filtered before application to the observer. In the case when the spectral densities of the signal and the noise are known, one may design an optimum filter minimizing the mean-square error, on the lines of the Wiener filter. This may be either an analogue filter, or a digital filter easily implemented on a microcomputer.

In the absence of prior knowledge of the statistics of the signal and the noise, one may use Widrow's adaptive filter utilizing the steepest descent method to seek the optimal solution. The configuration of the filter is shown in Fig. 4, where $w \in R^L$ is the adaptive filter weighting vector, chosen in such a manner as to minimize (in the scalar case)

$$J = E \{ [y(k) - \hat{y}(k)] [y(k) - \hat{y}(k)] \} \quad \text{for all } k \quad (16)$$

if the output of the filter is

$$\hat{y}_k = w^T(k) Y(k) = Y^T(k) w(k) \quad (17)$$

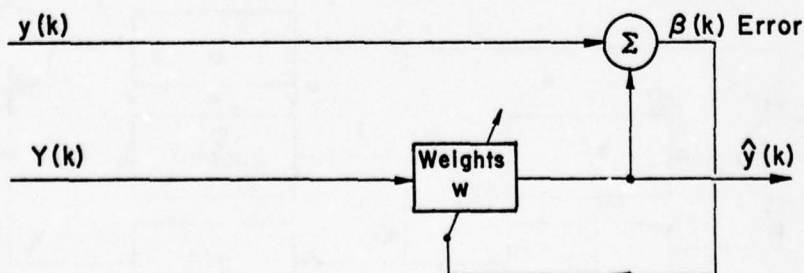


Fig. 4: Configuration of Widrow's filter.

and

$$Y(k) = [y(k-1) \ y(k-2) \ \dots \ y(k-l)]^T \quad (18)$$

The weighting vector, w , is updated at each iteration according to the relationship

$$w(k+1) = w(k) + 2c \beta(k) Y(k) \quad (19)$$

where

$$\beta(k) = y(k) - \hat{y}(k) \quad (20)$$

$$c_{\max} > c > 0 \quad (21)$$

and

$$c_{\max} = \text{trace } E [y(k) y^T(k)] \quad (22)$$

The implementation of Widrow's filter is shown in Fig. 5.

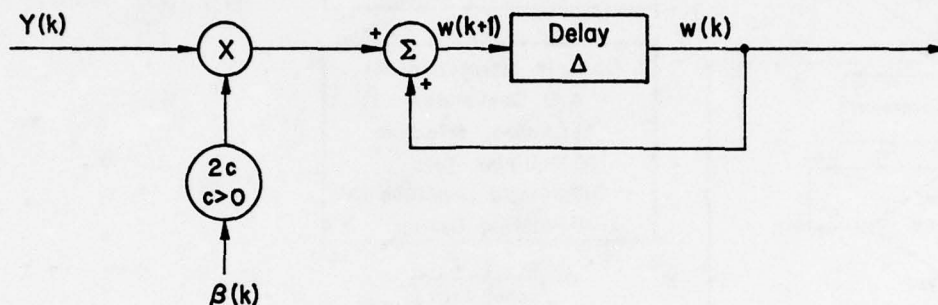


Fig. 5: Implementation of Widrow's filter.

RESULTS OF SIMULATION

To demonstrate the feasibility of the adaptive observer for on-line state estimation, first a second-order linear single-input single-output time-invariant system was simulated on a TR-20 analogue computer. The system is described by the following equations

$$\dot{x}(t) = \begin{bmatrix} -0.4 & 1 \\ 0 & -0.2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \quad (23)$$

$$y(t) = [1 \ 0] x(t) \quad (24)$$

Since the state $x_1(t)$ is directly observed as the output $y(t)$, our problem is to estimate the state $x_2(t)$.

The INTEL MDS-800 microcomputer was used to process the input and the output from the analogue computer after analogue-to-digital conversion. Since the poles of the system are located at -0.4 and -0.2 , a sampling period of 0.7 second was selected, taking into consideration the on-line processing time. For the discretized system, the following state transition equation was obtained

$$x(k+1) = \begin{bmatrix} 0.755784 & 0.567872 \\ 0 & 0.869358 \end{bmatrix} x(k) + \begin{bmatrix} 0.213341 \\ 0.653209 \end{bmatrix} u(k) \quad (25)$$

For this linear case, a first-order observer is easily obtained, described by the following equations

$$z(k+1) = -3.09222 u(k) + 0.5 z(k) + y(k) \quad (26)$$

$$\hat{x}_2(k) = 0.650425 y(k) - 0.166308 z(k) \quad (27)$$

where the pole of the observer was placed at 0.5 .

Equations (26) and (27) were implemented on the microcomputer with the SBC-310 Mathematical Unit Board [6] to speed up the arithmetical operations as compared to a complete software realization. The programmes were written in the 8080 assembly language [7] with hard-wired interrupt handling capability. The flowchart of the programme is shown in Fig. 6 and the flowchart of the data flow is shown in Fig. 7.

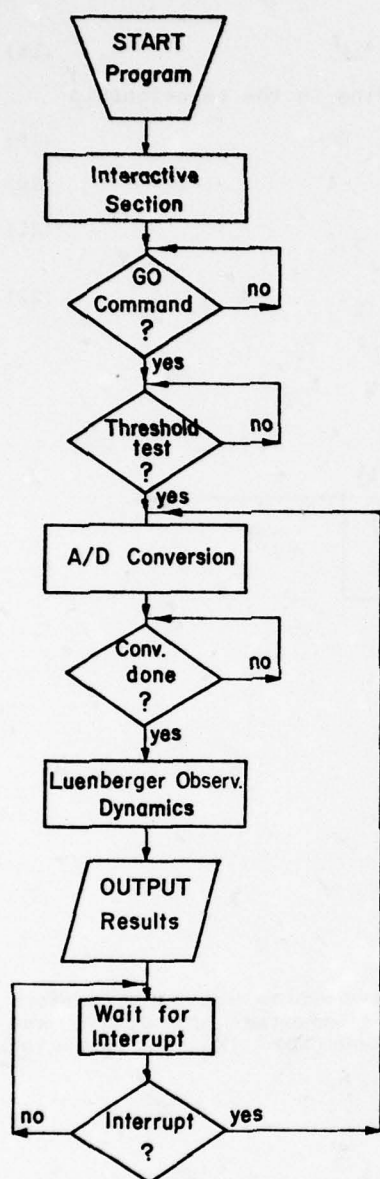


Fig. 6: Flow Chart

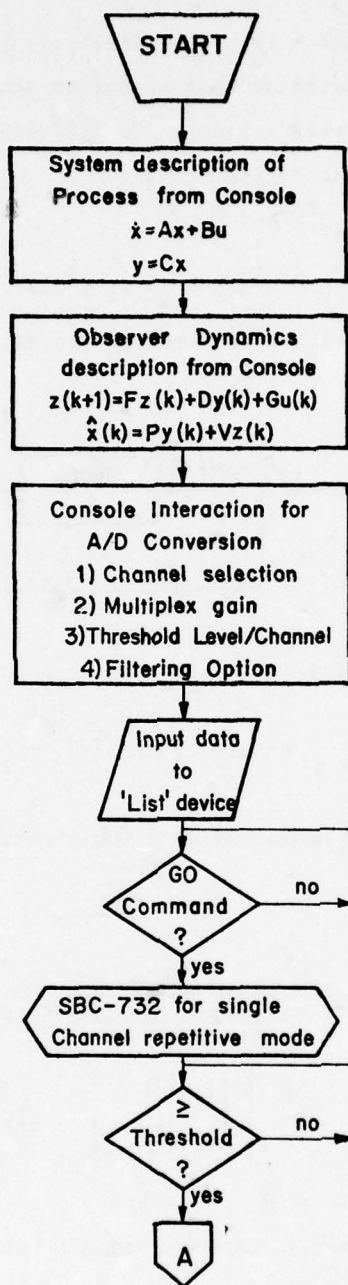
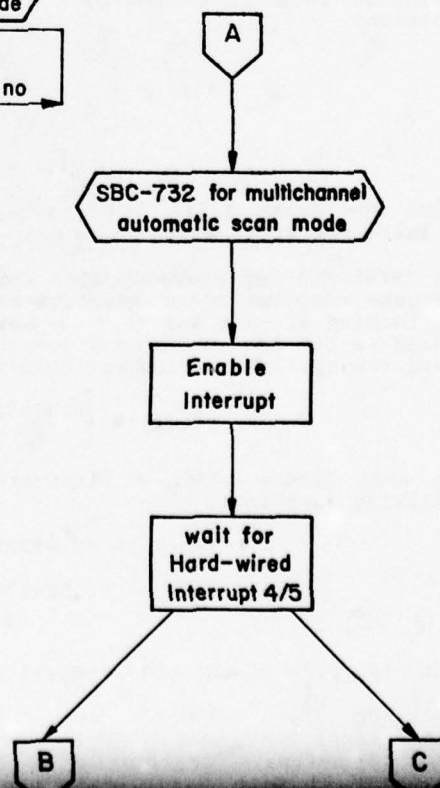


Fig. 7(a): Detailed Flow Chart

Fig. 7(b): Detailed Flow Chart Continued



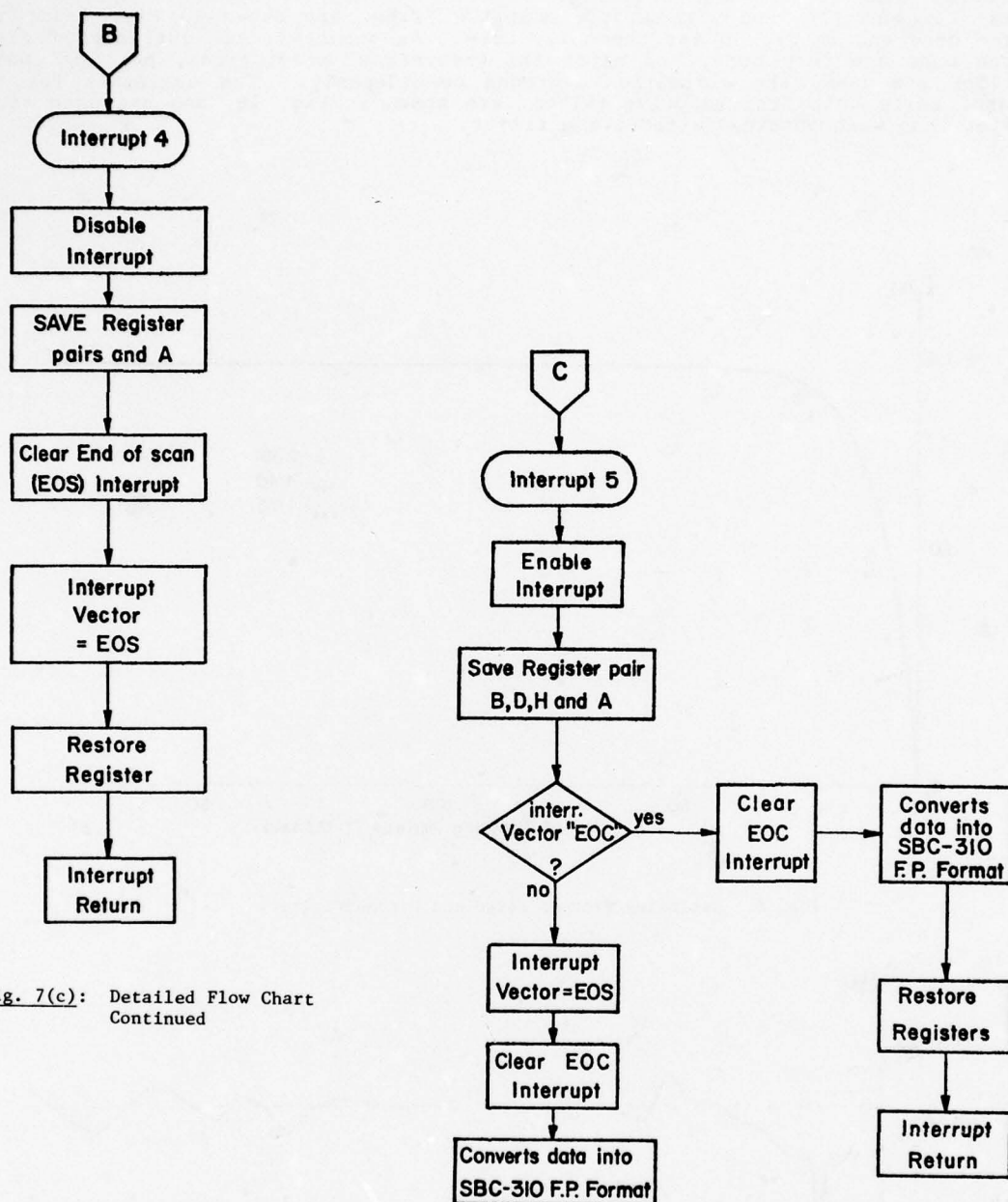
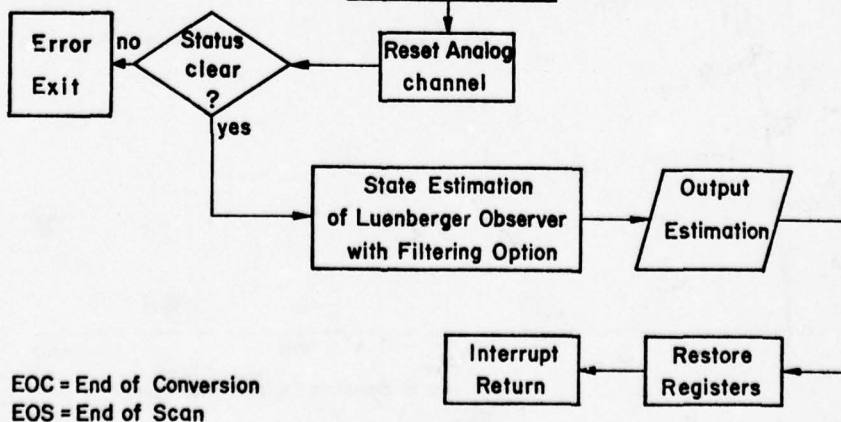


Fig. 7(c): Detailed Flow Chart Continued



EOC = End of Conversion
EOS = End of Scan

Fig. 7(d): Detailed Flow Chart Continued

For the purpose of simulation, a step input was used. Two runs were conducted, one without external noise added to the output, and the other with external noise injected from a noise generator. The estimates obtained using the observer indicated by equations (26) and (27) and without the adaptive filter are shown in Fig. 8 for the noise-free case and in Fig. 9 for the noisy case. As expected, the estimates for the noise-free case are very good, and match the theoretical predictions, but when noisy observations are used, the estimation degrades considerably. The estimates for the noisy case, while using the adaptive filter, are shown in Fig. 10, and are seen to be much better than when obtained without the filter.

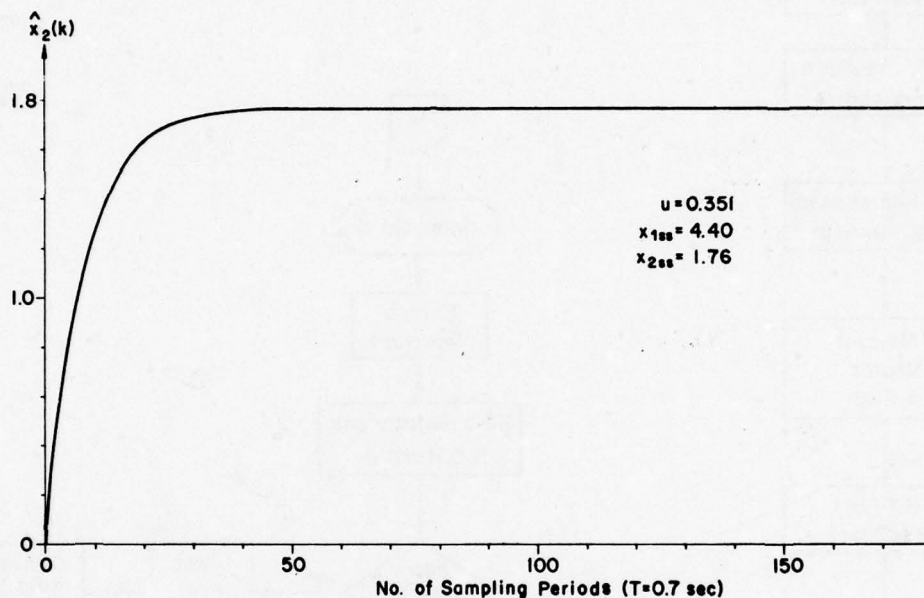


Fig. 8: Estimates without noise and without filter.

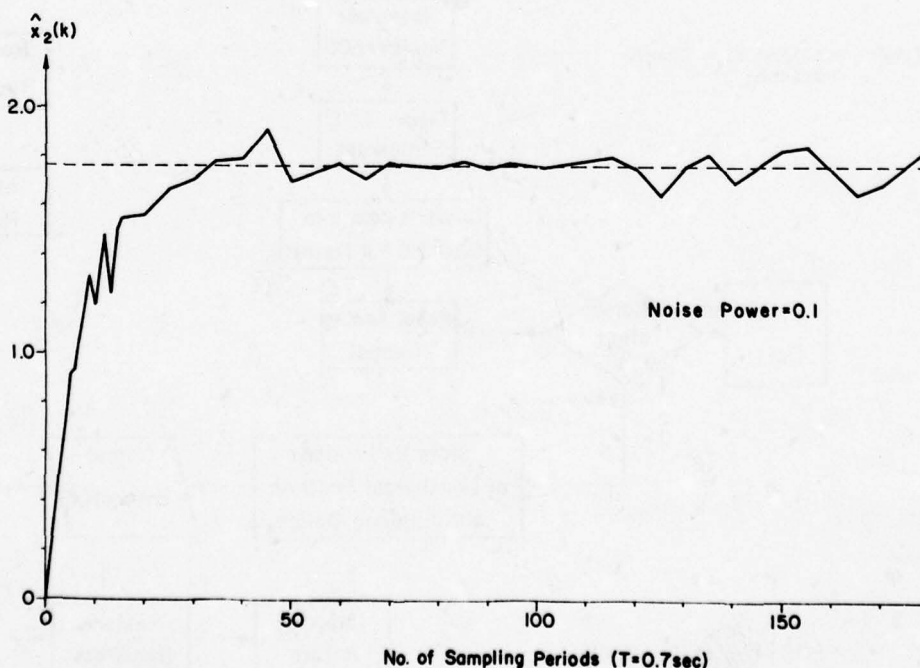


Fig. 9: Estimates with noise but without filter.

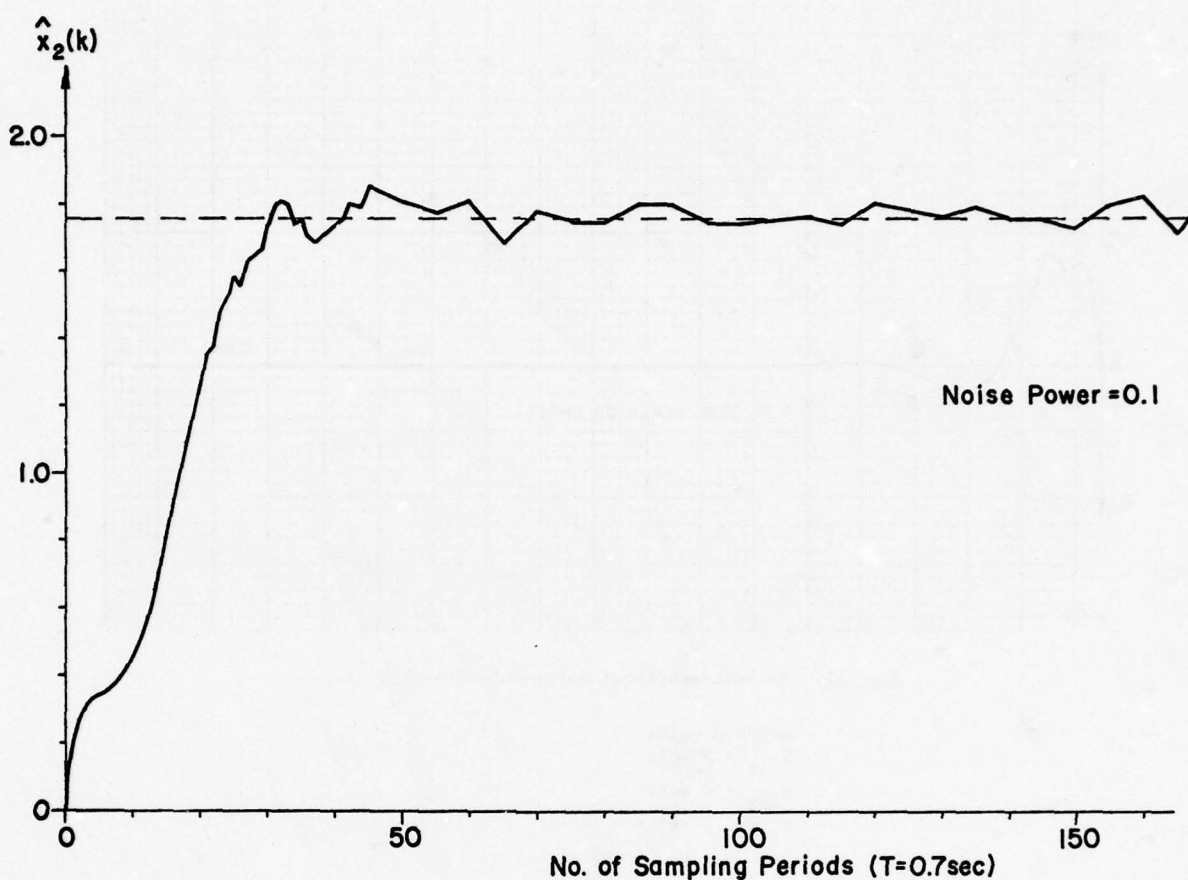


Fig. 10: Estimates with noise and filter.

Next, a second-order nonlinear system was considered. This system is described by the equations

$$\left. \begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -3x_1 - 2x_1^2 - x_2 + u \end{aligned} \right\} \quad (28)$$

and

$$y = x_1 \quad (29)$$

Linearizing the system about $x_1 = 0$, $x_2 = 0$, a second-order observer was designed, given by the following equations

$$\left. \begin{aligned} \dot{\hat{x}}_1 &= \hat{x}_2 - \hat{x}_1 + y \\ \dot{\hat{x}}_2 &= -2\hat{x}_1^2 - \hat{x}_2 + u + 3y \end{aligned} \right\} \quad (30)$$

The linearized observer has poles at $s = -1$.

For the purpose of simulation, a step input was again used. The estimates obtained for the noise-free case, with the observer alone, are shown in Fig. 11. When 0.2-volt r.m.s. noise was added, the estimates without filter are shown in Fig. 12, whereas the estimates with a fourth-order digital Chebyshev filter (cut-off frequency 0.2 Hz) are shown in Fig. 13. It will be seen that the nonlinear observer with the filter works quite well even in the presence of noise.

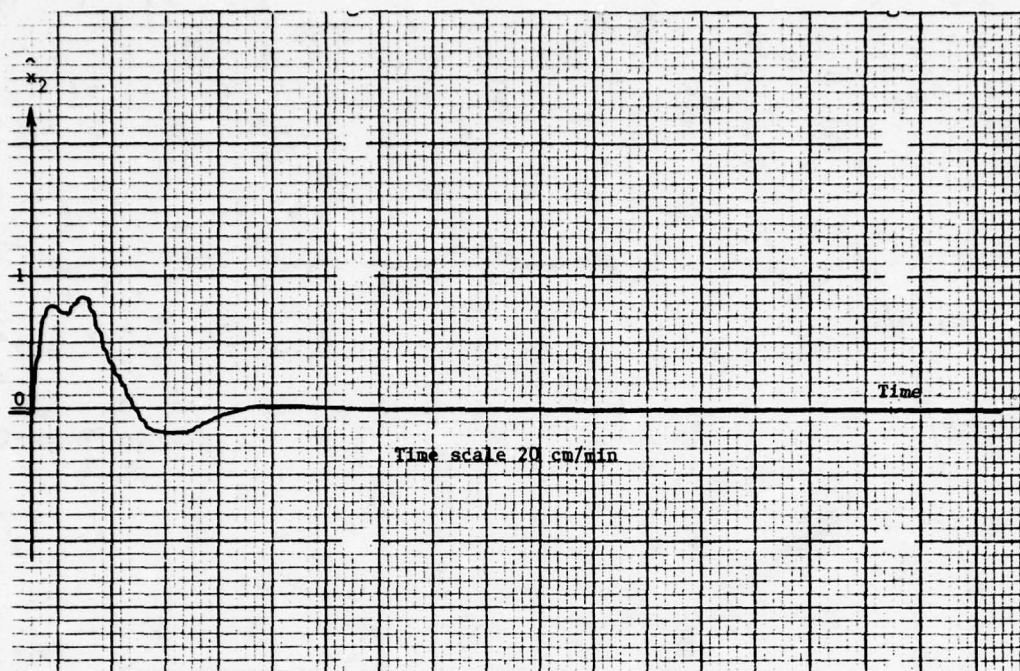


Fig. 11: Estimates without noise and without filter

$u = 5.0$ volts
 $x_{1ss} = 1$ volt
 $x_{2ss} = 0$ volt
 $\sigma_v = 0.2$ volts

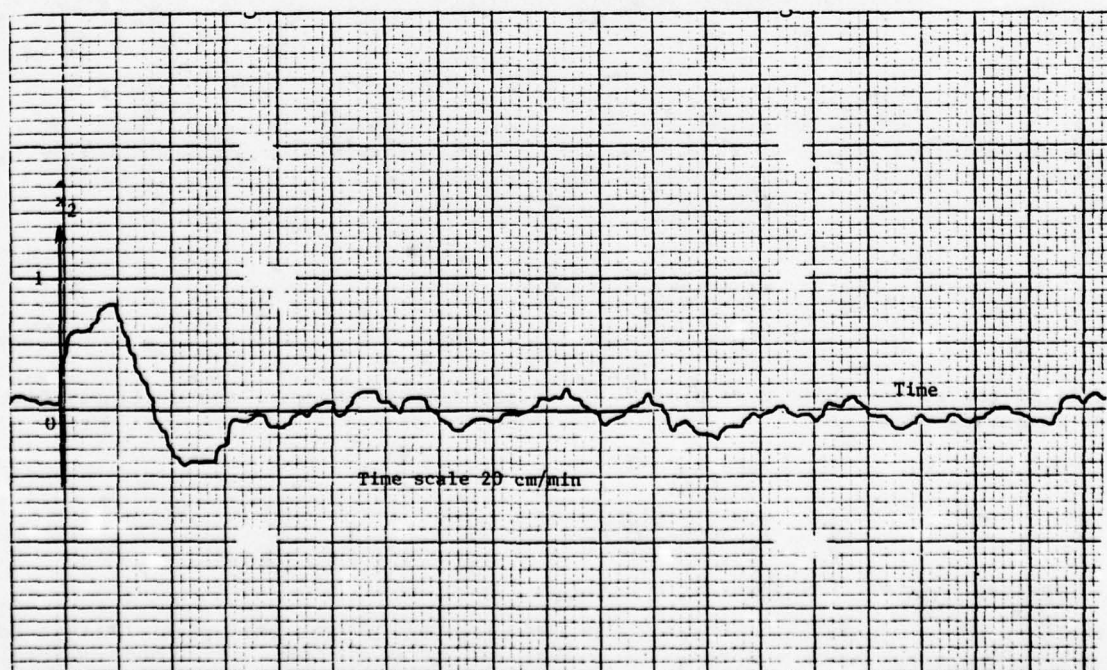


Fig. 12: Estimates with noise but without filter.

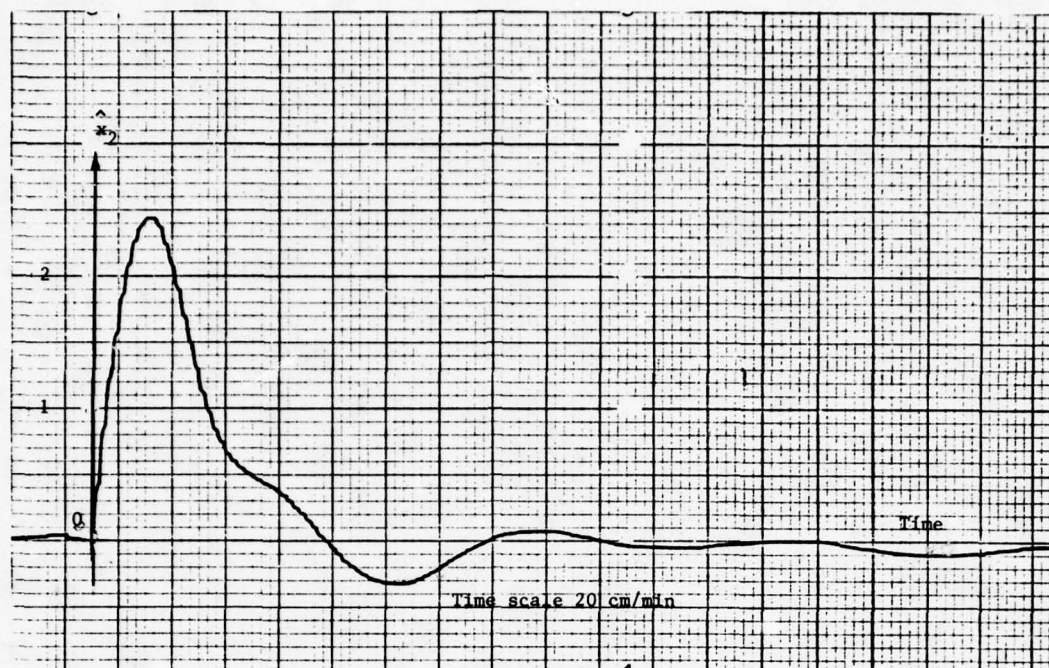


Fig. 13: Estimates with noise and fixed filter.

CONCLUSIONS

It has been shown that the proposed scheme for estimating the states of a nonlinear system works well for a simulated second-order system. With high-order systems, the observer must be made adaptive, in order to ensure stability.

This can be done by dividing the state space into several regions, for each of which a given value of the elements of the observer matrix L will give a specified degree of stability. These values may be pre-calculated, and stored in the memory of the microcomputer to be used in the form of a look-up table.

It remains to verify the proposed scheme for the adaptive observer for the case of a communications satellite. This is currently under investigation.

ACKNOWLEDGEMENT

The support of this work by the Communications Research Centre under Research Contract No. OSU78-00069 is gratefully acknowledged.

REFERENCES

- [1] N.K. Sinha and S.A. Azim, "Adaptive estimation of the states of a synchronous orbit satellite", *Int. J. Systems Science*, vol. 9, 1978, pp. 121-127.
- [2] N.K. Sinha and A.F.W. Tom, "Adaptive state estimation for systems with coloured observation noise", *Int. J. Systems Science*, vol. 9, 1978, pp. 31-43.
- [3] S.P. Altman, "A unified state model of orbital trajectory and attitude dynamics", *Celestial Mechanics*, vol. 6, 1972, pp. 452-496.
- [4] S.A. Azim and N.K. Sinha, "Comparison of different methods of nonlinear state estimation with applications to a satellite", *Proc. 16th Annual Allerton Conference on Communication, Control and Computing (Monticello, Illinois)*, October 1978, pp. 202-212.
- [5] B. Widrow, J.R. Gliver, Jr., J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeidler, E. Dong, Jr., and R.C. Goodlin, "Adaptive noise cancelling: principles and applications", *Proc. IEEE*, vol. 63, 1975, pp. 1692-1716.
- [6] Intel Corporation, SBC-310 High Speed Mathematics Unit Hardware Reference Manual, 1977.
- [7] Intel Corporation, Intel 8080 Microcomputer System Users' Manual, 1975.

METHODS FOR STRAP-DOWN ATTITUDE ESTIMATION AND NAVIGATION WITH ACCELEROMETERS

PROF. IR. R.P. OFFEREINS, IR. M.J.L. TIERNEGO

TWENTE UNIVERSITY OF TECHNOLOGY
 DEPT. OF ELECTRICAL ENGINEERING
 P.O. BOX 217
 7500 AE ENSCHEDE, THE NETHERLANDS

SUMMARY

The paper describes methods for calculating the attitude of a vehicle from the signals of three linear and three angular accelerometers which are rigidly attached to the vehicle. Also course, velocity and position measurements relative to some object can be used. Apart from the attitude also the velocity and position with respect to this object are obtained as output signals. In fire control systems filters for target position prediction and attitude determination can be combined in this way.

LIST OF SYMBOLS

X, Y, Z	position coordinates with respect to local vertical system
x, y, z	position coordinates with respect to ship-fixed system
M	transformation matrix relating x, y, z to X, Y, Z
α, β, γ	attitude angles
ω_E, R, λ	earth rotation rate, earth radius, latitude
$V_X, V_Y, V_Z, V_N, V_E, V_H, V_L$	velocity components in X, Y, Z , north, east, upward and forward direction
$a_x, a_y, a_z, a_X, a_Y, a_Z$	acceleration components in x, y, z, X, Y and Z direction
t_x, t_y, t_z	measured acceleration components
ρ_x, ρ_y, ρ_z	angular acceleration components
r_x, r_y, r_z	measured angular acceleration components
$\omega_x, \omega_y, \omega_z$	angular velocity components with respect to the XYZ-system
$\omega_{xs}, \omega_{ys}, \omega_{zs}$	angular velocity components with respect to space
v_{tx}, v_{ty}, v_{tz}	errors in linear acceleration measurements
v_{rx}, v_{ry}, v_{rz}	errors in angular acceleration measurements
P_t	spectral density of linear acceleration measurement error
P_r	spectral density of angular acceleration measurement error
P_b	spectral density of derivative of angular accelerometer bias
w_x, w_y, w_z	errors in position measurements
w_K, w_L	errors in course and forward velocity measurements
q_r	spectral density of position errors
q_k, q_v	spectral density of course, resp. forward velocity measuring error
A, B, C	system matrix, control matrix, measurement matrix
P, Q	covariance matrices for resp. input noise vector and measurement noise vector
R, r_{ij}	error covariance matrix R with element r_{ij}
k_1 to k_9	filter feedback coefficients
ω_0	filter bandwidth

Symbols with \sim represent estimated variables.

Symbols with Δ represent differences between actual and estimated variables.

1. INTRODUCTION

Usually the estimation of a vehicles attitude with respect to the local vertical is based on a stabilized platform with gyros and linear accelerometers, which is an expensive mechanical precision construction. Systems with no platform, where the acceleration or velocity sensors are attached to the vehicle are called "strap-down" systems (ref. 2). In this paper methods are described to estimate the attitude from three angular and three linear accelerometers rigidly attached to the vehicle. The signals of these accelerometers are processed by a computer. The basic idea is that a double integration of angular acceleration results in the angle and that also linear accelerometer signals contain information about the attitude due to the effect of gravity. Ref. 1 describes a combination of a low-pass filter processing linear accelerometer signals and a high-pass filter processing angular accelerometer signals, together resulting in information about the angle. Internal reports (ref. 6, 7, 8) considered various aspects of this system.

In this paper a more general approach is chosen based on the Kalman filter concept (ref. 3, 4, 5). Then also other sensors can be easily added to the system. In section 2 the general Kalman filter approach and its equations, which are used in later sections, are given.

Section 3 gives various equations relating acceleration, attitude, effect of earth rotation and curvature. These equations are used in three examples, which are worked out in detail.

The first most simple one which is described in section 4 and 5 uses six accelerometers and a course measurement. The second one of section 6 and 7 introduces also velocity measurements. In this example the effect of earth rotation and curvature is included. In fact this example could be used as part of a navigation system. The third example of section 8 and 9 uses as a sensor also position measurements relative to some external object moving with constant velocity. The practical use of this example could be for tracking purposes and for fire control systems.

In all examples first the model is described and then the filter. Models and filters are given in block diagram form. The filter parameters (feedback coefficients) are calculated using the equations of section 2. The models are subdivided in loosely coupled subsystems. For these subsystems the calculations are carried out. The systems are described as continuous-time systems. For practical use the filters have to be discretized. Section 10 indicates how the attitude limitation present in the methods of the first sections can be eliminated. The more extensive formula manipulation is done in the appendix. For convenience a list of the most important symbols used is given at the beginning of the paper.

2. KALMAN FILTER CONCEPT

In fig. 1 the principle of the Kalman filter is shown. A model of an actual system generates output signals which are compared with the actual output signals of the system. The differences of these two are used to correct the state variables of the model. The input signals entering the actual system are also input signals of the model. The mathematical equations describing the system are as follows:

Actual system equations:

$$\begin{aligned}\frac{dx}{dt} &= A x + B u + v \\ y &= C x + w\end{aligned}\quad (1)$$

x , u , y , v and w respectively represent state vector, input vector, output vector, input noise vector and output noise vector. A , B and C are matrices.

Filter equations including model and correction:

$$\begin{aligned}\frac{d\hat{x}}{dt} &= A \hat{x} + B u + K (y - \hat{y}) \\ \hat{y} &= C \hat{x}\end{aligned}\quad (2)$$

\hat{x} and \hat{y} are the estimated state vector and output vector. The noise vectors v and w , which are supposed to be white noise vectors, are characterized by their covariance matrices P and Q . The elements p_{ij} and q_{ij} multiplied by the delta function $\delta(\tau)$ represent the expected values of the products $E\{v_i(\tau) \cdot v_j(\tau)\}$ and $E\{w_i(\tau) \cdot w_j(\tau)\}$.

If no correlation exists between two different noise signals then $p_{ij} = 0$ and $q_{ij} = 0$ for $i \neq j$. The diagonal elements p_{ii} and q_{ii} represent the magnitude of the power density spectrum of the noise signals v_i and w_i respectively. Kalman filter theory shows that the optimal value of K is found from the equations.

$$\begin{aligned}K &= R C' Q^{-1} \\ \frac{dR}{dt} &= A R + R A' - R C' Q^{-1} C R + P\end{aligned}\quad (3)$$

Optimality means here that the error variances $E\{\Delta x_i^2\} = E\{(x_i - \hat{x}_i)^2\}$ are minimal. This optimal value K can be found from the error model which describes the behaviour of $\Delta x = x - \hat{x}$ and $\Delta y = y - \hat{y}$. This error model follows from (1) and (2)

$$\begin{aligned}\frac{d\Delta x}{dt} &= A \Delta x + v - K \Delta y \\ \Delta y &= C \Delta x + w\end{aligned}\quad (4)$$

The unknown input noise vector v in fig. 1 is acting on the actual system. There are situations as in this paper that the input to the actual system is measured and that these measurements, contaminated with noise, are used as inputs in the filter model. Then the same optimal filter is valid. The above equations (1) and (2) are linear equations. For non-linear systems an optimal filter is found also by making a model of this system and correcting this model. For the differences between actual system and model we can use a linearized error model satisfying (4) in which the various matrices now may depend on the state \hat{x} of the filter model.

3. EQUATIONS RELATING ACCELERATION AND ATTITUDE

In this section we derive various equations relating linear and angular acceleration to ships attitude including the effect of earth curvature and earth rotation. These equations are used in the models and filters of the next sections.

We consider a ship-fixed xyz-coordinate system with z-axis perpendicular to the deck, x-axis pointing forward and y-axis pointing starboard and an XYZ-coordinate system with Z-axis pointing local vertical upward, X-axis pointing north and Y-axis pointing east. The origins of the two systems coincide.

Three angular accelerometers and three linear accelerometers are attached to the xyz-system in the origin. They measure the angular accelerations of the xyz-system around the x, y and z-axis and the linear accelerations in x, y and z-direction with respect to space.

Coordinates of a point in the xyz and XYZ-systems are related by a transformation matrix \underline{M} according to the equation

$$\underline{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underline{M} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \underline{M} \underline{X} \quad (5)$$

The elements m_{11} , m_{12} and m_{13} are the cosines of the angles between z-axis and resp. X, Y and Z-axis, the elements m_{21} , m_{22} and m_{23} are the cosines of the angles between y-axis and resp. X, Y and Z-axis, the elements m_{31} , m_{32} and m_{33} are the cosines of the angles between x-axis and resp. X, Y and Z-axis. These nine elements are not mutually independent. There are six relations between the elements due to the fact that \underline{M} is orthonormal, which is expressed as

$$\underline{M} \underline{M}' = \underline{I} \quad (6)$$

where \underline{M}' is the transpose of \underline{M} . This means physically that in both coordinate systems the coordinate axes are mutually perpendicular and the same unit of length is used.

There are various possibilities to choose three independent angles. The so-called Euler angles are obtained, if starting from a situation that the two coordinate systems coincide, the system is rotated successively around the z-axis, the y-axis and the x-axis again. A mechanical visualisation of these Euler angles is shown in fig. 2. The inner frame represents the vehicle.

In this paper the three angles as shown in the mechanical model of fig. 3 are used, where again the inner frame represents the vehicle. The xyz-system can be given an arbitrary rotation with respect to the XYZ-system starting from coincidence, as shown in fig. 3, by rotating it successively the angle γ around the z-axis, the angle β around the y-axis and the angle α around the x-axis. The elements of \underline{M} can be expressed in α , β and γ by respectively multiplying the transformation matrices belonging to the rotations γ , β and α .

$$\underline{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha \\ 0 & -\sin \alpha & \cos \alpha \end{bmatrix} \begin{bmatrix} \cos \beta & 0 & -\sin \beta \\ 0 & 1 & 0 \\ \sin \beta & 0 & \cos \beta \end{bmatrix} \begin{bmatrix} \cos \gamma & \sin \gamma & 0 \\ -\sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} \cos \beta \cos \gamma & \cos \beta \sin \gamma & -\sin \beta \\ \sin \alpha \sin \beta \cos \gamma - \cos \alpha \sin \gamma & \sin \alpha \sin \beta \sin \gamma + \cos \alpha \cos \gamma & \sin \alpha \cos \beta \\ \cos \alpha \sin \beta \cos \gamma + \sin \alpha \sin \gamma & \cos \alpha \sin \beta \sin \gamma - \sin \alpha \cos \gamma & \cos \alpha \cos \beta \end{bmatrix} \quad (7)$$

For rotating systems \underline{M} is time dependent

$$\frac{d\underline{M}}{dt} = \begin{bmatrix} \dot{m}_{11} & \dot{m}_{12} & \dot{m}_{13} \\ \dot{m}_{21} & \dot{m}_{22} & \dot{m}_{23} \\ \dot{m}_{31} & \dot{m}_{32} & \dot{m}_{33} \end{bmatrix} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} = \underline{\Omega} \underline{M} \quad (8)$$

ω_x , ω_y , ω_z are the angular velocities of the xyz-system around the x, y and z-axis with respect to the XYZ-system. From (7) and (8) it follows that

$$\begin{aligned} \frac{d\alpha}{dt} &= \omega_x + \tan \beta (\omega_y \sin \alpha + \omega_z \cos \alpha) \\ \frac{d\beta}{dt} &= \omega_y \cos \alpha - \omega_z \sin \alpha \\ \frac{d\gamma}{dt} &= \frac{1}{\cos \beta} (\omega_y \sin \alpha + \omega_z \cos \alpha) \end{aligned} \quad (9)$$

The values ω_x , ω_y , ω_z are calculated from the angular velocities ω_{xs} , ω_{ys} , ω_{zs} of the xyz-system with respect to space by subtracting the velocity components due to earth rotation and earth curvature.

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \omega_{xs} \\ \omega_{ys} \\ \omega_{zs} \end{bmatrix} - \underline{M} \begin{bmatrix} (\omega_E + \frac{V_E}{R \cos \lambda}) \cos \lambda \\ -\frac{V_N}{R} \\ (\omega_E + \frac{V_E}{R \cos \lambda}) \sin \lambda \end{bmatrix} \quad (10)$$

where ω_E , V_N , V_E , λ and R respectively represent earth rotation rate, north pointing velocity component of 0, east pointing velocity component of 0, latitude of 0 and earth radius.

The angular acceleration components ρ_x , ρ_y and ρ_z are related to the angular velocities ω_{xs} , ω_{ys} , ω_{zs} by

$$\begin{aligned} \frac{d\omega_{xs}}{dt} &= \rho_x \\ \frac{d\omega_{ys}}{dt} &= \rho_y \\ \frac{d\omega_{zs}}{dt} &= \rho_z \end{aligned} \quad (11)$$

The linear acceleration components a_x , a_y and a_z of 0 in x, y and z direction with respect to space including the effect of gravity as measured by the accelerometers are related to the acceleration components in xyz-direction:

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \underline{M} \begin{bmatrix} a_X \\ a_Y \\ a_Z \end{bmatrix} + \underline{M} \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} \quad (12)$$

or

$$\begin{bmatrix} a_X \\ a_Y \\ a_Z \end{bmatrix} = \underline{M}^{-1} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} \quad (13)$$

In (12) and (13) G represents the gravity constant. These acceleration components a_x , a_y and a_z with respect to space in X, Y and Z direction can be transformed (see appendix 11.1) to acceleration components a_N , a_E and a_H in X, Y and Z (= north, east, upward) direction with respect to the XYZ-system by

$$\begin{aligned} a_N &= a_X - 2(\omega_E + \frac{V_E}{R \cos \lambda}) V_E \sin \lambda - 2 \frac{V_N V_H}{R} - \frac{V_N^2}{R} + (\omega_E + \frac{V_E}{R \cos \lambda})^2 R \cos \lambda \sin \lambda \\ a_E &= a_Y + 2(\omega_E + \frac{V_E}{R \cos \lambda}) V_N \sin \lambda - 2(\omega_E + \frac{V_E}{R \cos \lambda}) V_H \cos \lambda \\ a_H &= a_Z + 2(\omega_E + \frac{V_E}{R \cos \lambda}) V_E \cos \lambda + 2 \frac{V_N^2}{R} - (\omega_E + \frac{V_E}{R \cos \lambda})^2 R \cos^2 \lambda \end{aligned} \quad (14)$$

V_H is the velocity component in Z-direction. The equations derived above are used in the various models and filters of the next sections. In the filters we use the same equations as in the models, however with the actual variables replaced by estimated variables, which are denoted with the same symbols only with a $\hat{\cdot}$, for example $\hat{\alpha}$ and \hat{a} . Reference numbers to corresponding equations with estimated variables have a $\hat{\cdot}$, so (9)' is the same equation as (9) however with estimated variables. In many cases the accuracies involved are such that the effect of earth rotation and earth curvature can be neglected. Then in (10) and (14) $\omega_E = 0$ and $R = \infty$, so

$$\begin{aligned} \omega_X &= \omega_{xs} & a_N &= a_X \\ \omega_Y &= \omega_{ys} & a_E &= a_Y \\ \omega_Z &= \omega_{zs} & a_H &= a_Z \end{aligned} \quad (15)$$

4. MODEL FOR ATTITUDE GENERATION FROM ACCELERATION

The model for attitude generation from angular acceleration is shown in fig. 4. As input signals the angular accelerations ρ_x , ρ_y , ρ_z are used. They are integrated to angular velocity components ω_x , ω_y and ω_z around x, y and z-axis. Earth rotation and earth curvature are neglected. The time derivatives of α , β and γ are determined from ω_x , ω_y and ω_z by (9). α , β and γ are obtained by integration.

The input variables ρ_x , ρ_y and ρ_z are measured by angular accelerometers. These measurements are supposed to be biased by bias signals b_x , b_y and b_z which are obtained as output signals from integrators and are added in the model to the input variables ρ_x , ρ_y and ρ_z . Noise signals v_{bx} , v_{by} and v_{bz} , each with power density spectrum p_b , account for bias drift.

Noise signals v_{rx} , v_{ry} and v_{rz} each with power density spectrum p_r are added to represent the measurement noise in the measured angular accelerations r_x , r_y and r_z . Linear accelerations $t_x = a_x$, $t_y = a_y$ and $t_z = a_z$ are measured and considered as measurement signals for the attitude angles α , β and γ according to (12). In this equation the second term in the right hand side is the attitude measuring part and the first term representing the acceleration components of the origin 0 is considered as measurement noise. We may neglect measurement noise due to inaccuracies of the accelerometers because it is considered to be small as compared with the linear acceleration of 0.

5. FILTER FOR ATTITUDE DETERMINATION FROM ACCELERATION MEASUREMENTS

The filter is obtained as described in section 2. The result is shown in fig. 5. It is seen by comparing fig. 4 and fig. 5, that the filter of fig. 5 contains the model of fig. 4. In this section a linearized error model is determined. This error model is subdivided into three third order subsystems. By neglecting the coupling between these subsystems it is possible to calculate the feedback matrix \underline{K} for these subsystems independently. The values of the feedback coefficients obtained in this way are indicated in fig. 5.

In the following equations differences between actual variables and corresponding variables are denoted by Δ , so $\Delta \alpha = \alpha - \hat{\alpha}$. As can be seen from fig. 4 the only non-linear equations in the model are (9) and the measurement equation (12). Linearizing (9) around a reference path, for which we chose the path followed by the estimated variables, results in

$$\begin{aligned} \frac{d\Delta \alpha}{dt} &= \Delta \omega_x + \Delta \omega_y \tan \hat{\beta} \sin \hat{\alpha} + \Delta \omega_z \tan \hat{\beta} \cos \hat{\alpha} \\ &\quad + \Delta \alpha \tan \hat{\beta} (\hat{\omega}_y \cos \hat{\alpha} - \hat{\omega}_z \sin \hat{\alpha}) - \Delta \beta \frac{1}{\cos^2 \hat{\beta}} (\hat{\omega}_y \sin \hat{\alpha} + \hat{\omega}_z \cos \hat{\alpha}) \\ \frac{d\Delta \beta}{dt} &= \Delta \omega_y \cos \hat{\alpha} - \Delta \omega_z \sin \hat{\alpha} - \Delta \alpha (\hat{\omega}_y \sin \hat{\alpha} + \hat{\omega}_z \cos \hat{\alpha}) \end{aligned}$$

$$\frac{d\Delta\gamma}{dt} = \Delta\omega_z \frac{\cos\hat{\alpha}}{\cos\hat{\beta}} + \Delta\omega_y \frac{\sin\hat{\alpha}}{\cos\hat{\beta}} - \Delta\alpha \frac{\hat{\omega}_y \cos\hat{\alpha} - \hat{\omega}_z \sin\hat{\alpha}}{\cos\hat{\beta}} - \Delta\beta \frac{\sin\hat{\beta}}{\cos^2\hat{\beta}} (\hat{\omega}_y \sin\hat{\alpha} + \hat{\omega}_z \cos\hat{\alpha}) \quad (16)$$

The linear acceleration signals t_x , t_y and t_z are used as error correcting signals after multiplication by \hat{M}^{-1} . From (12) we get

$$\hat{M}^{-1} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \hat{M}^{-1} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \hat{M}^{-1} \underline{M} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \hat{M}^{-1} \underline{M} \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} \quad (17)$$

For small values $\Delta\alpha$, $\Delta\beta$ and $\Delta\gamma$ we have

$$\hat{M}^{-1} \underline{M} = \begin{bmatrix} 1 & \Delta\gamma & -\Delta\beta \\ -\Delta\gamma & 1 & \Delta\alpha \\ \Delta\beta & -\Delta\alpha & 1 \end{bmatrix} \quad (18)$$

We consider the unknown acceleration components a_x , a_y and a_z as measurement noise so we can write

$$\hat{M}^{-1} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \begin{bmatrix} -G\Delta\beta + \text{noise} \\ G\Delta\alpha + \text{noise} \\ G + \text{noise} \end{bmatrix} \quad (19)$$

The linearized error model obtained as described is 9th order. It can be subdivided into three subsystems, the α -system with state variables $\Delta\omega_x$, $\Delta\alpha$ and Δb_x , the β -system with state variables $\Delta\omega_y$, $\Delta\beta$ and Δb_y and the γ -system with state variables $\Delta\omega_z$, $\Delta\gamma$ and Δb_z . The three subsystems are coupled by (16).

In order to get simple results, we neglect the coupling and use for the calculation of the feedback matrix \underline{K} instead of (16) the equation (20):

$$\begin{aligned} \frac{d\Delta\alpha}{dt} &= \Delta\omega_x \\ \frac{d\Delta\beta}{dt} &= \Delta\omega_y \\ \frac{d\Delta\gamma}{dt} &= \Delta\omega_z \end{aligned} \quad (20)$$

This neglect can be used only if the values $\hat{\alpha}$, $\hat{\beta}$ and $\frac{d\hat{\gamma}}{dt} = \hat{\omega}_y \sin\hat{\alpha} + \hat{\omega}_z \cos\hat{\alpha}$ are sufficiently small.

Using the symbols of section 2 the three subsystems are characterized by the state matrix

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

The α -system has as measurement matrix $\underline{C} = [0 \ 0 \ G]$

The β -system has as measurement matrix $\underline{C} = [0 \ 0 \ -G]$

As shown in fig. 3 the course γ is measured by the compass, so the measurement matrix \underline{C} for the γ -system is $\underline{C} = [0 \ 0 \ 1]$. The matrix \underline{P} for the system noise for each subsystem is

$$\underline{P} = \begin{bmatrix} p_b & 0 & 0 \\ 0 & p_r & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where p_b is the spectral density of the noise signals v_{bx} , v_{by} and v_{bz} and p_r is the spectral density of the noise signals v_{rx} , v_{ry} , v_{rz} .

The measurement noise matrix for the α -system and the β -system is $[Q] = [q_a]$ where q_a is the spectral density of the acceleration components of 0.

For the γ -system $[Q] = [q_k]$ where q_k is the spectral density of the course measurement error w_k .

The linearized error models characterized by the matrices mentioned above are used in the appendix to calculate the feedback matrices \underline{K} from (3) for the stationary case with $\dot{R} = 0$. The values of the feedback coefficients are shown in fig. 4. The accuracy is calculated as

$$\begin{aligned} E\{\Delta\alpha^2\} &= E\{\Delta\beta^2\} = 2p_r^{1/4} q_a^{3/4} G^{-3/2} \\ E\{\Delta\gamma^2\} &= 2p_r^{1/4} q_k^{3/4} \end{aligned} \quad (21)$$

The transfer function of the linearized filter relating angular acceleration and attitude angle for the α -system and the β -system is

$$\frac{1}{s^2} \frac{s^3}{s^3 + \sqrt{2} \omega_0 s^2 + \omega_0^2 s + \lambda \omega_0^2} = \frac{1}{s^2} H_1(s) \quad (22)$$

The factor $\frac{1}{s^2}$ represents the double integration to obtain angle from angular acceleration. The factor $H_1(s)$ represents a high pass filter, which means that only for high frequencies the angles can be obtained from angular acceleration.

The transfer function of the linearized filter relating linear acceleration and attitude angle is

$$-\frac{1}{G} \frac{\sqrt{2} \omega_0 s^2 + \omega_0^2 s + \lambda \omega_0^2}{s^3 + \sqrt{2} \omega_0 s^2 + \omega_0^2 s + \lambda \omega_0^2} = -\frac{1}{G} H_2(s) \quad (23)$$

$H_2(s)$ is a low pass filter showing that for low frequencies the attitude angle is obtained from the linear accelerometer signal. Note that $H_1(s) + H_2(s) = 1$, which means that contributions of angular and linear accelerometer to the angle value together give the right angle.

6. MODEL FOR ATTITUDE AND VELOCITY GENERATION FROM ACCELERATION

The model used in this section is shown in fig. 6. The attitude angles are determined from angular accelerations in a similar way as in section 4 and as shown in fig. 4. The difference is that in this example we do not neglect earth curvature and earth rotation. This means that the angular velocities ω_{xs} , ω_{ys} and ω_{zs} of the xyz-system with respect to space have to be transformed using (10) to angular velocities ω_x , ω_y and ω_z with respect to the XYZ-system.

In this model the linear accelerations as measured by the linear accelerometers are not used as measurement signals for the angles α , β and γ , but they act as inputs in the system. They are transformed by (13) to acceleration components a_x , a_y and a_z . (14) accounts for earth rotation and curvature and transforms these acceleration signals to acceleration signals a_N , a_E and a_H in north, east and upward direction. They are integrated to velocity components V_N , V_E and V_H in north, east and upward direction. V_N and V_E can be transformed to velocity components V and V_D .

$$\begin{aligned} V &= V_N \cos \gamma + V_E \sin \gamma \\ V_D &= -V_N \sin \gamma + V_E \cos \gamma \end{aligned} \quad (24)$$

The latitude λ to be used in (10) and (14) can be found by integrating V_N/R .

It is supposed that the forward velocity V is measured and that these measurements are contaminated with noise w_L , so the measured velocity is $V_L = V + w_L$. It is supposed that the velocity V_D is measured and that this measurement is zero. The actual unknown variations in V_D are accounted for by measurement noise $w_D = -V_D$. The input variables a_x , a_y and a_z are measured by the linear accelerometers. Measurement noise signals v_{tx} , v_{ty} and v_{tz} , each with spectral density p_t are added.

7. FILTER FOR ATTITUDE AND VELOCITY DETERMINATION FROM ACCELERATION, COURSE AND VELOCITY MEASUREMENTS

The filter is again obtained as described in section 2. The result is shown in fig. 7. It contains the model of fig. 6. In this section a linearized error model is determined. This error model is subdivided into three subsystems. By neglecting the coupling between the subsystems it is possible to calculate the feedback matrices K for the subsystems. In the error model earth rotation and earth curvature can be neglected, so (10) and (14) are omitted. The remaining non linear equations in the model are (9), (13) and the measurement equation (24). Linearization of (9) resulted in (16).

As control signals in the filter we use the measured linear accelerations t_x , t_y and t_z , which are obtained from a_x , a_y and a_z by adding measurement noise so

$$\begin{aligned} t_x &= a_x + v_{tx} \\ t_y &= a_y + v_{ty} \\ t_z &= a_z + v_{tz} \end{aligned} \quad (25)$$

They are transformed to estimated acceleration components \hat{a}_x , \hat{a}_y and \hat{a}_z in X, Y and Z direction by (26).

$$\begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix} = \hat{M}^{-1} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} \quad (26)$$

From (12), (25) and (26) we get

$$\begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix} = \hat{M}^{-1} \underline{M} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \hat{M}^{-1} \underline{M} \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} + \hat{M}^{-1} \begin{bmatrix} v_{tx} \\ v_{ty} \\ v_{tz} \end{bmatrix} \quad (27)$$

Together with (18) we get

$$\begin{aligned}\hat{a}_X &= a_X + \Delta\gamma a_Y - \Delta\beta a_Z - G\Delta\beta + v'_{tx} \\ \hat{a}_Y &= a_Y - \Delta\gamma a_X + \Delta\alpha a_Z + G\Delta\alpha + v'_{ty} \\ \hat{a}_Z &= a_Z + \Delta\beta a_Y - \Delta\alpha a_X + v'_{tz}\end{aligned}\quad (28)$$

The linear error model equations are

$$\begin{aligned}\Delta a_X &= a_X - \hat{a}_X = G\Delta\beta - \Delta\gamma \hat{a}_Y + \Delta\beta \hat{a}_Z - v'_{tx} \\ \Delta a_Y &= a_Y - \hat{a}_Y = -G\Delta\alpha + \Delta\gamma \hat{a}_X - \Delta\alpha \hat{a}_Z - v'_{ty} \\ \Delta a_Z &= a_Z - \hat{a}_Z = -\Delta\beta \hat{a}_Y + \Delta\alpha \hat{a}_X - v'_{tz}\end{aligned}\quad (29)$$

The non linear error signals are

$$\begin{aligned}\Delta V_L &= V_L - \hat{V}_L = V_N \cos\gamma + V_E \sin\gamma - (\hat{V}_N \cos\hat{\gamma} + \hat{V}_E \sin\hat{\gamma}) + w_L \\ \Delta V_D &= V_D - \hat{V}_D = -V_N \sin\gamma + V_E \cos\gamma - (-\hat{V}_N \sin\hat{\gamma} + \hat{V}_E \cos\hat{\gamma}) + w_D\end{aligned}\quad (30)$$

These error signals are transformed to error correcting signals e_N and e_E

$$\begin{aligned}e_N &= \Delta V_L \cos\hat{\gamma} - \Delta V_D \sin\hat{\gamma} \\ e_E &= \Delta V_L \sin\hat{\gamma} + \Delta V_D \cos\hat{\gamma}\end{aligned}\quad (31)$$

(30) and (31) result in

$$\begin{aligned}e_N &= V_N \cos\Delta\gamma + V_E \sin\Delta\gamma - \hat{V}_N + w'_L \\ e_E &= -V_N \sin\Delta\gamma + V_E \cos\Delta\gamma - \hat{V}_E + w'_D\end{aligned}\quad (32)$$

$$\begin{aligned}\text{with } w'_L &= w_L \cos\hat{\gamma} - w_D \sin\hat{\gamma} \\ w'_D &= w_L \sin\hat{\gamma} + w_D \cos\hat{\gamma}\end{aligned}$$

For small values of $\Delta\gamma$ the linearized error model is

$$\begin{aligned}e_N &= \Delta V_N + w'_L + V_E \Delta\gamma \\ e_E &= \Delta V_E + w'_D - V_N \Delta\gamma\end{aligned}\quad (33)$$

The system can be subdivided into three subsystems, the α -system with state variables Δb_x , $\Delta\omega_x$, $\Delta\alpha$ and ΔV_N ; the β -system with state variables Δb_y , $\Delta\omega_y$, $\Delta\beta$ and ΔV_N and the γ -system with state variables Δb_z , $\Delta\omega_z$ and $\Delta\gamma$. The three systems are coupled through (16) and (29). The coupling is neglected by using (20) instead of (16) and by using (34) and (35) instead of (29) and (33)

$$\begin{aligned}\Delta a_X &= G\Delta\beta + v''_{tx} \\ \Delta a_Y &= -G\Delta\alpha + v''_{ty}\end{aligned}\quad (34)$$

$$\begin{aligned}e_N &= \Delta V_N + w'_L \\ e_E &= \Delta V_E + w'_D\end{aligned}\quad (35)$$

The neglected terms representing the interaction can be considered as an increase in system noise from v'_{tx} , v'_{ty} , w'_L and w'_D to v_{tx}'' , v_{ty}'' , w_L'' and w_D'' . The matrices used in the appendix for calculating the feedback matrix \underline{K} using (3) for the various subsystems are given below.

α -system: state variables Δb_x , $\Delta\omega_x$, $\Delta\alpha$, ΔV_N

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -G & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_b & 0 & 0 & 0 \\ 0 & p_r & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_t \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \quad \underline{Q} = \begin{bmatrix} q_v \end{bmatrix}\quad (36)$$

β -system: state variables Δb_y , $\Delta\omega_y$, $\Delta\beta$, ΔV_N

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & G & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_b & 0 & 0 & 0 \\ 0 & p_r & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_t \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \quad \underline{Q} = \begin{bmatrix} q_v \end{bmatrix}\quad (37)$$

Y-system: state variables $\Delta b_z, \Delta \omega_z, \Delta \gamma$

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_b & 0 & 0 \\ 0 & p_r & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \quad \underline{Q} = \begin{bmatrix} q_k \end{bmatrix} \quad (38)$$

For a relatively small value of p_r the values k_1 till k_7 are

$$\begin{aligned} k_1 &= G^{-1} \lambda \omega_o^3 & k_2 &= G^{-1} \omega_o^3 & k_3 &= 2G^{-1} \omega_o^2 & k_4 &= 2\omega_o \\ \text{with } \omega_o &= p_r^{1/6} q_v^{-1/6} G^{1/3} & \text{and } \lambda &= p_b^{1/2} p_r^{-1/2} \\ k_5 &= \lambda \omega_Y^2 & k_6 &= \omega_Y^2 & k_7 &= 2^{1/2} \omega_Y \\ \text{with } \omega_Y &= p_r^{1/4} q_k^{-1/4} & \text{and } \lambda &= p_b^{1/2} p_r^{-1/2} \end{aligned} \quad (39)$$

The accuracy of estimation is given by

$$\begin{aligned} E\{\Delta \alpha^2\} &= E\{\Delta \beta^2\} = 3p_r^{1/2} q_v^{1/2} C^{-1} \\ E\{\Delta \gamma^2\} &= \sqrt{2} p_r^{1/4} q_k^{-3/4} & E\{\Delta V_L^2\} &= 2p_r^{1/6} q_v^{5/6} G^{1/3} \end{aligned} \quad (40)$$

The transfer function for the decoupled linearized α - and β -systems relating angular acceleration and attitude angle are

$$\frac{1}{s^2} \frac{s^4 + 2\omega_o s^3}{s^4 + 2\omega_o s^3 + 2\omega_o s^2 + \omega_o^3 s + \lambda \omega_o^3} = \frac{1}{s^2} H_1(s) \quad (41)$$

The factor $\frac{1}{s^2}$ represents the double integration to obtain angle from angular acceleration. The factor $H_1(s)$ represents a high pass filter. The transfer functions of the linearized filter relating linear acceleration and attitude angle are

$$\frac{1}{G} \frac{2\omega_o^2 s^2 + \omega_o^3 s + \lambda \omega_o^3}{s^4 + 2\omega_o s^3 + 2\omega_o s^2 + \omega_o^3 s + \lambda \omega_o^3} = \frac{1}{G} H_2(s) \quad (42)$$

$H_2(s)$ is a low pass filter. Note that $H_1(s) + H_2(s) = 1$.

The contribution of the angular accelerometer to the angle value in the high frequency region and the contribution of the linear accelerometer to the angle value in the low frequency region together give the right angle. The "separation" frequency defined as the intersection of the high frequency -12 db/oct asymptote of $H_2(s)$ and the low frequency +12 db/oct asymptote of $H_1(s)$, that is obtained by putting the small value $\lambda = 0$, is ω_o .

8. MODEL FOR ATTITUDE, VELOCITY AND POSITION GENERATION FROM ACCELERATION, COURSE AND POSITION MEASUREMENTS

The model used in this section is shown in fig. 8. Earth rotation and earth curvature are neglected. Attitude angles are generated from angular acceleration in the same way as in the model of fig. 4 and fig. 6. As in former sections the measured accelerations r_x, r_y and r_z are supposed to be biased and contaminated with noise. Linear accelerations are used as input signals. They are measured and these measurements are supposed to be contaminated with noise.

As in the example of fig. 6 the linear acceleration components a_x, a_y and a_z in x, y and z direction, including the effect of gravity, are transformed with (13) to components a_X, a_Y and a_Z in X, Y and Z direction. They are integrated to velocity components V_X, V_Y and V_Z . The velocity components are integrated to position coordinates X, Y and Z .

The velocity components and position coordinates are relative to an arbitrary external object moving with constant velocity. This object could be some known fixed object, or, as in fire control systems, it could be the target. In fire control systems, it is necessary to predict the target position and for this prediction an assumption has to be made regarding velocity in the near future and the most obvious assumption is a constant velocity.

Unknown acceleration components of the object can be accounted for by extra input noise, so by increasing the spectral density of the noise signals v_{tx}, v_{ty} and v_{tz} . Known acceleration components of the object could be added to the acceleration components of the vehicle. The position coordinates X, Y and Z are transformed to position coordinates x, y and z by

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underline{M} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (43)$$

These position coordinates are measured, for example by radar.

There are noise signals w_x , w_y and w_z , each with spectral density q_r , so the measured position coordinates are

$$\begin{aligned} x_r &= x + w_x \\ y_r &= y + w_y \\ z_r &= z + w_z \end{aligned} \quad (44)$$

9. FILTER FOR ATTITUDE, VELOCITY AND POSITION DETERMINATION FROM ACCELERATION, COURSE AND POSITION MEASUREMENTS

The filter is obtained as described in section 2 from the model of section 8. The result is shown in fig. 9. In this section a linearized error model is determined. This error model is subdivided into four subsystems. By neglecting the coupling between these subsystems it is possible to calculate the feedback matrices K for the subsystems. The values of the feedback coefficients k_i as indicated in fig. 9 are obtained in this way. The non linear equations in the model are (9), (13) and the measurement equation (43). Equations (9) and (13) have been linearized in previous sections. They changed to (16) and (29). As regards the non linear measurement equation (43) we proceed as follows. The error correction signals e_x , e_y and e_z used in fig. 9 are

$$\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \hat{M}^{-1} \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} - \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} \quad (45)$$

With (44) and (45) we get

$$\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \hat{M}^{-1} \underline{M} \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} + \hat{M}^{-1} \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} \quad (46)$$

With (18) this becomes

$$\begin{aligned} e_x &= \Delta X - \Delta\beta\hat{z} + w_x' + \Delta\gamma\hat{y} \\ e_y &= \Delta Y + \Delta\alpha\hat{z} + w_y' - \Delta\gamma\hat{x} \\ e_z &= \Delta Z + w_z' + \Delta\beta\hat{x} - \Delta\alpha\hat{y} \end{aligned} \quad (47)$$

We consider four subsystems, the αY -system with state variables $\Delta\alpha$, $\Delta\omega_x$, Δb_x , ΔY and ΔV_y , the βX -system with state variables $\Delta\beta$, $\Delta\omega_y$, Δb_y , ΔX and ΔV_x , the γ -system with state variables $\Delta\gamma$, $\Delta\omega_z$ and Δb_z and the Z -system with state variables ΔZ and ΔV_z .

Decoupling of the four subsystems is done by using (20) instead of (16) and (34) instead of (29) as in the case of section 7. For the βX -system we use as error correcting signal e_x of (47) and we account for the influence of the γ -system via $\Delta\gamma$ by increasing the measurement noise.

In the same way we use for the αY -system the error correcting signal e_y of (47) and account for the influence of the γ -system by increasing the measurement noise. The signal e_z of (47) is used as error correcting signal for the Z -system. The influence of the βX -system and the αY -system via $\Delta\beta$ and $\Delta\alpha$ is accounted for by increasing the measurement noise. As error correcting signal for the γ -system we use the difference $K - \hat{y}$ as in the previous cases.

The matrices to be used in the appendix for calculating the feedback matrices K for the subsystems according to (3) for the various systems are:

αY -system: state variables Δb_x , $\Delta\omega_x$, $\Delta\alpha$, ΔV_y , ΔY

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -G & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_b & 0 & 0 & 0 & 0 \\ 0 & p_r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_t & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 0 & \hat{z} & 0 & 1 \end{bmatrix} \quad (48)$$

$$\underline{Q} = \begin{bmatrix} q_r \end{bmatrix}$$

βX -system: state variables Δb_y , $\Delta\omega_y$, $\Delta\beta$, ΔV_x , ΔX

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & G & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_b & 0 & 0 & 0 & 0 \\ 0 & p_r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_t & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 0 & -\hat{z} & 0 & 1 \end{bmatrix} \quad (49)$$

$$\underline{Q} = \begin{bmatrix} q_r \end{bmatrix}$$

γ -system: state variables Δb_z , $\Delta\omega_z$, $\Delta\gamma$

$$\underline{A} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_b & 0 & 0 \\ 0 & p_r & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \quad (50)$$

$$\underline{Q} = \begin{bmatrix} q_k \end{bmatrix}$$

Z-system: state variables $\Delta V_Z, \Delta Z$

$$\underline{A} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \quad \underline{P} = \begin{bmatrix} p_t & 0 \\ 0 & 0 \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad \underline{Q} = \begin{bmatrix} q_r \end{bmatrix} \quad (51)$$

The values of the various feedback coefficients of the filter of fig. 10 as calculated in the appendix 11.4 are:

$$\begin{aligned} k_1 &= G^{-1} \lambda \omega_o^4 & \text{with } \lambda &= p_b^{1/2} p_r^{-1/2} \\ k_2 &= G^{-1} \omega_o^4 & \omega_o &= G^{1/4} p_r^{1/8} q_r^{-1/8} \\ k_3 &= 2^{3/4} \mu^{1/2} G^{-1} \omega_o^3 & \omega_Y &= p_r^{1/4} q_k^{-1/4} \\ k_4 &= 2^{1/2} \mu \omega_o^2 & \omega_Z &= p_t^{1/4} q_r^{-1/4} \\ k_5 &= 2^{-3/4} \mu^{-1/2} (1-a-b\mu+\mu^2) \omega_o & a &= 2^{-1} G^{-1} p_r^{-1/2} q_r^{-1/2} p_t \\ k_6 &= \lambda \omega_Y^2 & b &= 2^{1/2} \hat{z} G^{-1/2} p_r^{1/4} q_r^{-1/4} \\ k_7 &= \omega_Y^2 & \mu &= f(a,b) \quad (\text{see table}) \\ k_8 &= 2^{1/2} \omega_Y & & \\ k_9 &= \omega_Z^2 & & \\ k_{10} &= 2^{1/2} \omega_Z & & \end{aligned} \quad (52)$$

Table for $\mu(a,b)$

a \ b	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
0	0,24	0,31	0,41	0,62	1	1,61	2,41	3,30	4,24	5,19	6,16	7,14	8,12	9,11
1	0,66	0,79	0,96	1,22	1,60	2,14	2,83	3,63	4,49	5,40	6,34	7,29	8,26	9,23
2	0,92	1,08	1,29	1,58	1,98	2,50	3,14	3,90	4,72	5,60	6,51	7,44	8,39	9,34
3	1,14	1,32	1,56	1,87	2,27	2,79	3,41	4,14	4,93	5,78	6,67	7,58	8,51	9,46
4	1,34	1,54	1,79	2,12	2,53	3,04	3,65	4,35	5,12	5,95	6,82	7,71	8,63	9,57
5	1,52	1,73	2	2,34	2,75	3,26	3,86	4,55	5,30	6,11	6,96	7,84	8,75	9,67
6	1,68	1,91	2,19	2,54	2,96	3,47	4,05	4,73	5,47	6,26	7,10	7,97	8,86	9,78
7	1,84	2,08	2,37	2,72	3,15	3,66	4,24	4,90	5,63	6,41	7,24	8,09	8,98	9,88
8	1,99	2,24	2,54	2,90	3,33	3,83	4,41	5,07	5,78	6,55	7,36	8,21	9,08	9,98
9	2,13	2,39	2,70	3,06	3,50	4	4,58	5,22	5,93	6,69	7,49	8,33	9,19	10,08
10	2,27	2,54	2,85	3,22	3,65	4,16	4,73	5,37	6,07	6,82	7,61	8,44	9,30	10,17

The quantities a , b and μ are dimensionless. p_t and \hat{z} occur only in resp. a and b .

The accuracy of the estimated variables is given by the elements r_{15} of \underline{R} . The accuracy of the angle determination is

$$E(\Delta \alpha^2) = E(\Delta \beta)^2 = r_{33} \quad (53)$$

and the accuracy of a predicted target position X_p with prediction time T is

$$E(\Delta X_p^2) = E\{\Delta(X + TV_X)^2\} = E(\Delta X^2) + 2TE(\Delta X \cdot \Delta V_X) + T^2 E(\Delta V_X)^2 = r_{55} + 2Tr_{45} + T^2 r_{44} \quad (54)$$

The values r_{33} , r_{44} , r_{45} and r_{55} as calculated in the Appendix are

$$r_{33} = q_r \omega_o^5 G^{-2} 2^{-3/4} \mu^{-1/2} (3\mu^2 - b\mu - 1 + a) \quad (55)$$

$$r_{44} = q_r \omega_o^3 2^{-5/4} \mu^{-1/2} \{2\mu^3 - b\mu^2 + (b^2 - 2a - 2)\mu + b(1-a)\} \quad (56)$$

$$r_{45} = q_r \omega_o^2 2^{-1/2} (b\mu^2 + 2\mu - ab) \quad (57)$$

$$r_{55} = q_r \omega_o 2^{-7/4} \mu^{-1/2} \{(2+3b^2)\mu^2 + b(6-b^2)\mu + (1-a)(2-b^2)\} \quad (58)$$

For relatively accurate linear acceleration measurements and small values of \hat{z} ($a \ll 1$ and $b \ll 1$) we put $a = 0$, $b = 0$ and $\mu = 2,4$.

Then

$$E(\Delta \alpha^2) = 6,3 q_r G^{-2} \omega_o^5 \quad (59)$$

$$E(\Delta X_p^2) = q_r \omega_o (2,6 + 6,8 \omega_o T + 6,3 \omega_o^2 T^2) \quad (60)$$

The transfer functions of the linearized filters relating angular acceleration and angle for the decoupled αY - and βX -system are:

$$\frac{1}{s^2} \frac{s^5 + k_5 s^4 + k_4 s^3}{s^5 + k_5 s^4 + k_4 s^3 + Gk_3 s^2 + Gk_2 s + Gk_1} = \frac{1}{s^2} H_1(s) \quad (61)$$

As in the previous example $\frac{1}{s^2}$ represents the double integration necessary for obtaining angle from angular acceleration.

$H_1(s)$ is a high pass filter.

The transfer functions of the linearized filters relating linear acceleration and angle for the decoupled αY - and βX -system are:

$$\frac{k_3 s^2 + k_2 s + k_1}{s^5 + k_5 s^4 + k_4 s^3 + Gk_3 s^2 + Gk_2 s + Gk_1} = \frac{1}{G} H_2(s) \quad (62)$$

$H_2(s)$ is a low pass filter. As in the previous examples $H_1(s) + H_2(s) = 1$, so the sum of the low frequency part of the angle via $H_1(s)$ and of the high frequency part of the angle via $H_2(s)$ exactly equals one.

The separation frequency ω_s of the two filters defined as the intersection of the high frequency -18 db/oct asymptote of $H_2(s)$ and the low frequency +12 db/oct of $H_1(s)$, which is obtained by putting the small value $k_1 = 0$ is

$$\omega_s = \left(\frac{G^2 k_2 k_3}{k_4} \right)^{1/5} = 2^{1/20} \mu^{-1/10} \omega_0 \sim \omega_0 \quad (63)$$

The transfer functions relating linear acceleration to position X resp. Y for the βX - and αY -system are:

$$\frac{Gs^3}{s^5 + k_5 s^4 + k_4 s^3 + Gk_3 s^2 + Gk_2 s + Gk_1} = H_3(s) \quad (64)$$

The transfer functions relating linear acceleration to velocity V_X resp. V_Y for the βX - and αY -system are:

$$\frac{s^2 (s^2 + k_5 s + k_4)}{s^5 + k_5 s^4 + k_4 s^3 + Gk_3 s^2 + Gk_2 s + Gk_1} = H_4(s) \quad (65)$$

Transfer functions (62), (64) and (65) are also of interest to show the errors due to target acceleration a_t which was assumed zero in the model.

The angle error ϵ_β , the position error ϵ_x and the velocity error ϵ_v are related to the target acceleration a_t in X -direction by the transfer functions $H_2(s)$, $H_3(s)$ and $H_4(s)$.

For example the target acceleration a_t due to a course change ΔK with constant rate of change $\frac{dK}{dt}$ and constant forward velocity is part of a sinusoidal acceleration change a_t with angular frequency $\frac{dK}{dt}$ and amplitude $V \frac{dK}{dt}$ and the corresponding errors ϵ_β , ϵ_x and ϵ_v can be calculated resp. with $H_2(s)$, $H_3(s)$ and $H_4(s)$.

If the measured position coordinates x_r , y_r and z_r acting as inputs to the filter of fig. 9 are chosen equal to zero this means that the "external" object supposed to move with constant course and velocity is the vehicle itself (origin 0 of the xyz-system). This results in a filter for attitude determination without an external reference as in the first example of section 5, which can be used if the vehicle maintains constant course and velocity. The value q_r in (52) to (60) now represents the power density spectrum of the fluctuations of the position of 0 around an average path with constant course and velocity. For application in fire control systems where target position prediction and attitude determination are combined in the filter of fig. 9 the same filter structure can be used during periods with no target. The filter obtained in this way resembles the one described in ref. 1.

10. CONCLUDING REMARKS

The methods described in the previous sections are attitude limited. There is for $\beta = 90^\circ$ a singular point as can be seen from (9). This point corresponds with the gimbal lock situation for $\beta = 90^\circ$ from fig. 3. A second limitation arises from the fact that the decoupling in subsystems is only justified for angles α and β which are sufficiently small. This decoupling with the resulting simplification of the calculations is very desirable because only then the cycle time of the computer calculations can be reduced to values, that make practical on line implementation possible. Simulations must show to what magnitudes of α and β the methods of the previous sections can be used. It is felt that ranges for α and β from -45° to $+45^\circ$ will be possible. If this is true then the attitude limitation can be removed as follows. The formulas (9) and (14) show that the decoupling in subsystems for α values in the neighbourhood of $\alpha = 90^\circ$ is possible if the role of ω_x and ω_z are interchanged. In the various filters this is automatically done by introducing the correcting signals for α , β and γ before and not after the updating equations (9)'. The gimbal lock situation for $\beta = 90^\circ$ can be avoided by using Euler angles for ranges of β around this gimbal lock situation. Fig. 2 and fig. 3 show that the Euler angles have singular points at $\beta = 0^\circ$ and $\beta = 180^\circ$ and that the gimbal angles of this paper have singular points at $\beta = +90^\circ$ and $\beta = -90^\circ$. This means that around $\beta = +90^\circ$ and $\beta = -90^\circ$ Euler angles can be used. The methods of this paper can be worked out for Euler angles in a similar way, resulting in the same values of the feed back coefficients. Appropriate switching from the gimbal angles of fig. 3 to the Euler angles of fig. 2 and vice versa at angle values $\beta = +45^\circ$, -45° , $+135^\circ$ and -135° can be done. In the ranges where the gimbal angles of fig. 3 are used the Euler angles can be calculated from these gimbal angles and vice versa so that the initial conditions after switching are correct.

As a second remark we note that instead of angular accelerometers angular rate sensors can be used. The filters in that case become simpler because the integration of angular acceleration to angular velocity can be omitted.

11. APPENDIX

11.1 DERIVATION OF (14)

We consider two coordinate systems with coinciding origins. One is the local vertical XYZ-system of section 3, the second one is a not rotating $X_O Y_O Z_O$ -system. As shown in section 3 with (5) the relation between a point \underline{X} in the XYZ-system and a point \underline{X}_O in the $X_O Y_O Z_O$ -system is given by

$$\underline{X} = \underline{M} \underline{X}_O \quad (66)$$

where \underline{M} is the orthonormal transformation matrix. The rotation of the \underline{X} -system with respect to the \underline{X}_O -system results in a time dependent matrix \underline{M} .

$$\frac{d\underline{M}}{dt} = \underline{\Omega} \underline{M} \quad (67)$$

Here the elements of $\underline{\Omega}$ are the angular velocity components of the \underline{X} -system around the X, Y and Z axis due to earth rotation and motion of the origin along the earth surface. These angular velocity components around the X-axis, Y-axis and Z-axis are:

$$\underline{\Omega} = \begin{bmatrix} (\omega_E + \frac{V_E}{R \cos \lambda}) \cos \lambda, & -\frac{V_N}{R} & (\omega_E + \frac{V_E}{R \cos \lambda}) \sin \lambda \\ 0 & -(\omega_E + \frac{V_E}{R \cos \lambda}) \sin \lambda & -\frac{V_N}{R} \\ (\omega_E + \frac{V_E}{R}) \sin \lambda & 0 & -(\omega_E + \frac{V_E}{R}) \cos \lambda \\ +\frac{V_N}{R} & (\omega_E + \frac{V_E}{R \cos \lambda}) \cos \lambda & 0 \end{bmatrix} \quad (68)$$

Twice differentiating (66) results in

$$\underline{M} \frac{d^2 \underline{X}_O}{dt^2} = \frac{d^2 \underline{X}}{dt^2} - \underline{\Omega} \underline{X} \quad (69)$$

$$\underline{M} \frac{d^2 \underline{X}_O}{dt^2} = \frac{d^2 \underline{X}}{dt^2} - 2 \underline{\Omega} \frac{d\underline{X}}{dt} - \frac{d\underline{\Omega}}{dt} \underline{X} + \underline{\Omega}^2 \underline{X} \quad (70)$$

For the origin of the coordinate systems where the accelerometers are $\underline{X} = 0$, so

$$\underline{M} \frac{d^2 \underline{X}_O}{dt^2} = \frac{d^2 \underline{X}}{dt^2} - 2 \underline{\Omega} \frac{d\underline{X}}{dt} \quad (71)$$

The left hand side of (71) contains as elements the coriolis acceleration components of the origin 0 with respect to the \underline{X}_O -system in coordinates of the \underline{X} -system, due to rotation.

To get the acceleration components a_X , a_Y and a_Z in the \underline{X} -system with respect to space we have to add the centrifugal acceleration components, so

$$\begin{bmatrix} a_X \\ a_Y \\ a_Z \end{bmatrix} = \underline{M} \frac{d^2 \underline{X}_O}{dt^2} + \begin{bmatrix} \frac{V_N^2}{R} - (\omega_E + \frac{V_E}{R \cos \lambda})^2 R \cos \lambda \sin \lambda \\ 0 \\ (\omega_E + \frac{V_E}{R \cos \lambda})^2 R \cos^2 \lambda \end{bmatrix} \quad (72)$$

We substitute in (71)

$$\frac{d^2 \underline{X}}{dt^2} = \begin{bmatrix} a_N \\ a_E \\ a_H \end{bmatrix} \quad (73)$$

and

$$\frac{d\underline{X}}{dt} = \begin{bmatrix} V_N \\ V_E \\ V_Z \end{bmatrix} \quad (74)$$

and (68).

The resulting expression for $\underline{M} \frac{d^2 \underline{X}_O}{dt^2}$ is substituted in (72). The result is (14) of section 3.

11.2 CALCULATION FOR THE FILTER OF SECTION 5

The matrices \underline{A} , \underline{C} , \underline{P} and \underline{Q} of the α -system of section 5 are substituted in (3) with $\frac{d\underline{R}}{dt} = 0$. This results in six equations for the six independent elements r_{ij} of the symmetrix matrix \underline{R} .

$$\begin{aligned}
-G^2 r_{13}^2 + p_b q_a &= 0 \\
-G^2 r_{13} r_{23} + r_{11} q_a &= 0 \\
-G^2 r_{13} r_{33} + r_{12} q_a &= 0 \\
-G^2 r_{23}^2 + 2r_{12} q_a + p_r q_a &= 0 \\
-G^2 r_{23} r_{33} + r_{13} q_a + r_{22} q_a &= 0 \\
-G^2 r_{33}^2 + 2r_{23} q_a &= 0
\end{aligned} \tag{75}$$

For the β -system the only difference with the α -system is the matrix \underline{C} , where $-G$ is substituted for G , so we get the same equations (75).

From (75) it follows that $r_{13} = G^{-1} p_b^{1/2} q_a^{1/2}$. The bias drift variations are supposed to be small, so for solving r_{ij} for $i \neq 1$ we put $r_{1j} = 0$ in the last three equations of (75). In this way the values of the elements of \underline{R} and the feedback coefficients can easily be calculated.

For the γ -system we have to substitute 1 for G in (75) and the power density spectrum of the course measurements q_k for the power density spectrum of the acceleration q_a .

11.3 CALCULATIONS FOR THE FILTER OF SECTION 7

The system matrices \underline{A} , \underline{P} , \underline{C} and \underline{Q} of the β -system of section 7 substituted in (3) results with $\frac{dR}{dt} = 0$ in ten equations for the ten independent elements of the symmetric matrix \underline{R} .

$$\begin{aligned}
0 &= -r_{14}^2 + q_v p_b \\
0 &= -r_{14} r_{24} + q_v r_{11} \\
0 &= -r_{14} r_{23} + q_v r_{12} \\
0 &= -r_{14} r_{44} + q_v G r_{13} \\
0 &= -r_{24} r_{34} + q_v r_{13} + q_v r_{22} \\
0 &= -r_{24}^2 + 2q_v r_{12} + q_v p_r \\
0 &= -r_{24} r_{44} + q_v r_{14} + q_v G r_{23} \\
0 &= -r_{34}^2 + 2q_v r_{23} \\
0 &= -r_{34} r_{44} + q_v r_{24} + q_v G r_{33} \\
0 &= -r_{44}^2 + 2q_v G r_{34} + q_v p_t
\end{aligned} \tag{76}$$

From the first equation we find $r_{14} = p_b^{1/2} q_v^{1/2}$. The bias drift variations are supposed to be small, so p_b is small. For solving r_{ij} for $i \neq 1$ we may put $r_{1j} = 0$ in the last 6 equations. By eliminating variables these equations are reduced to

$$-\frac{r_{44}^4}{p_t^2 q_v^2} + 2 \frac{r_{44}^2}{p_t q_v} + \frac{r_{44}}{(p_t q_v)^{1/2}} \frac{8q_v G p_r^{1/2}}{p_t^{3/2}} - 1 = 0 \tag{77}$$

$$\text{Substituting } \frac{r_{44}}{(p_t q_v)^{1/2}} = z \text{ and } \frac{8G q_v p_r^{1/2}}{p_t^{3/2}} = d$$

$$\text{we get } z^4 - 2z^2 - dz + 1 = 0 \tag{78}$$

$$\text{or } d = \frac{(z^2 - 1)^2}{z} \tag{79}$$

For $d \gg 1$ is $z \gg 1$ and $z^3 = d$, so

$$r_{44} = (p_t q_v)^{1/2} d^{1/3} = 2G^{1/3} q_v^{5/6} p_r^{1/6} \tag{80}$$

The values of the other elements of \underline{R} and the values of the feedback coefficients can now be calculated. They are given in section 7.

The α -system differs from the β -system only in the element a_{43} of the matrix \underline{A} , which is $-G$ instead of G . The same equation (77) is obtained. The elements r_{14} , r_{24} and r_{34} of \underline{R} and the feedback coefficients k_1 , k_2 and k_3 change their signs.

The γ -system of section 7 is the same as the γ -system of section 5 and thus the calculations of 11.2 are valid.

11.4 CALCULATION FOR THE FILTER OF SECTION 9

The matrices \underline{A} , \underline{C} , \underline{P} and \underline{Q} of the β X-system of section 9 substituted in (3) with $\frac{dR}{dt} = 0$ results in 15 equations for the 15 independent elements of the symmetric matrix \underline{R} .

$$(r_{15} - \hat{z} r_{13})^2 = q_r p_b \tag{81.1}$$

$$(r_{15} - \hat{z} r_{13})(r_{25} - \hat{z} r_{23}) = q_r r_{11} \tag{81.2}$$

$$(r_{15} - \hat{z} r_{13}) (r_{35} - \hat{z} r_{33}) = q_r r_{12} \quad (81.3)$$

$$(r_{15} - \hat{z} r_{13}) (r_{45} - \hat{z} r_{34}) = q_r G r_{13} \quad (81.4)$$

$$(r_{15} - \hat{z} r_{13}) (r_{55} - \hat{z} r_{35}) = q_r r_{14} \quad (81.5)$$

$$(r_{25} - \hat{z} r_{23})^2 = 2q_r r_{12} + q_r p_r \quad (81.6)$$

$$(r_{25} - \hat{z} r_{23}) (r_{35} - \hat{z} r_{33}) = q_r (r_{13} + r_{22}) \quad (81.7)$$

$$(r_{25} - \hat{z} r_{23}) (r_{45} - \hat{z} r_{34}) = q_r (r_{14} + G r_{23}) \quad (81.8)$$

$$(r_{25} - \hat{z} r_{23}) (r_{55} - \hat{z} r_{35}) = q_r (r_{15} + r_{24}) \quad (81.9)$$

$$(r_{35} - \hat{z} r_{33})^2 = 2q_r r_{23} \quad (81.10)$$

$$(r_{35} - \hat{z} r_{33}) (r_{45} - \hat{z} r_{34}) = q_r (r_{24} + G r_{33}) \quad (81.11)$$

$$(r_{35} - \hat{z} r_{33}) (r_{55} - \hat{z} r_{35}) = q_r (r_{25} + r_{34}) \quad (81.12)$$

$$(r_{45} - \hat{z} r_{34})^2 = q_r (2G r_{34} + p_t) \quad (81.13)$$

$$(r_{45} - \hat{z} r_{34}) (r_{55} - \hat{z} r_{35}) = q_r (G r_{35} + r_{44}) \quad (81.14)$$

$$(r_{55} - \hat{z} r_{35})^2 = 2q_r r_{45} \quad (81.15)$$

Bias drift variations are supposed to be small, so p_b is small and for solving r_{ij} for $i \neq 1$ we put $r_{1j} = 0$ in (81.6) to (81.15).

We renumber (81.6), (81.10), (81.13) and (81.15) as

$$(r_{25} - \hat{z} r_{23})^2 = p_r q_r \quad (82.1)$$

$$(r_{35} - \hat{z} r_{33})^2 = 2q_r r_{23} \quad (82.2)$$

$$(r_{45} - \hat{z} r_{34})^2 = q_r (2G r_{34} + p_t) \quad (82.3)$$

$$(r_{55} - \hat{z} r_{35})^2 = 2q_r r_{45} \quad (82.4)$$

These values are substituted in the remaining equations after quadrating right and left hand sides

$$2p_r r_{23} = r_{22}^2 \quad (82.5)$$

$$p_r (2G r_{34} + p_t) = G^2 r_{23}^2 \quad (82.6)$$

$$2p_r r_{45} = r_{24}^2 \quad (82.7)$$

$$2r_{23} (2G r_{34} + p_t) = (G r_{33} + r_{24})^2 \quad (82.8)$$

$$4r_{23} r_{45} = (r_{25} + r_{34})^2 \quad (82.9)$$

$$2r_{45} (2G r_{34} + p_t) = (G r_{35} + r_{44})^2 \quad (82.10)$$

As from these 10 equations the 5 equations (82.1), (82.3), (82.6), (82.7) and (82.9) contain only the 5 variables r_{23} , r_{24} , r_{25} , r_{34} and r_{45} , they can be solved from these equations. First we eliminate r_{45} from (82.3) and (82.9) using (82.7) and we substitute in (82.3) the value $2G r_{34} + p_t$ from (82.6). This results in

$$(r_{25} - \hat{z} r_{23})^2 = p_r q_r \quad (83.1)$$

$$r_{24}^2 - 2 \hat{z} p_r r_{34} = 2G r_{23} p_r^{1/2} q_r^{1/2} \quad (83.2)$$

$$p_r (2G r_{34} + p_t) = G^2 r_{23}^2 \quad (83.3)$$

$$2r_{23} r_{24}^2 = p_r (r_{25} + r_{34})^2 \quad (83.4)$$

We eliminate r_{24}^2 from (83.2) and (83.4). This results in

$$(r_{25} + r_{34})^2 - 4 \hat{z} r_{23} r_{34} = 4G r_{23}^2 p_r^{-1/2} q_r^{1/2} \quad (84)$$

In this equation we substitute

$$r_{25} = p_r^{1/2} q_r^{1/2} + \hat{z} r_{23} \text{ and } r_{34} = \frac{1}{2} G r_{23}^2 p_r^{-1} - \frac{1}{2} p_t G^{-1}$$

as obtained from (83.1) and (83.3). The result is a 4th order equation in r_{23} :

$$r_{23}^4 G^2 p_r^{-2} - 4r_{23}^3 \hat{z} G p_r^{-1} + 4r_{23}^2 (-3G p_r^{-1/2} q_r^{1/2} - \frac{1}{2} p_r^{-1} p_t + \hat{z}^2) + 4r_{23} (2\hat{z} p_r^{1/2} q_r^{1/2} + \hat{z} G^{-1} p_t) + 4p_r q_r + G^{-2} p_t^2 - 4G^{-1} p_r^{1/2} q_r^{1/2} p_t = 0 \quad (85)$$

Dividing the left hand side by $4p_r q_r$ and then introducing the dimensionless variable u so, that

$$u^4 = r_{23}^4 \frac{G^2}{4p_r^3 q_r} \quad (86)$$

results in

$$\begin{aligned}
& u^4 - u^3 2^{3/2} \hat{z} G^{-1/2} p_r^{1/4} q_r^{-1/4} + \\
& u^2 (-6 - G^{-1} p_r^{-1/2} q_r^{-1/2} p_t + 2 \hat{z}^2 G^{-1} p_r^{1/2} q_r^{-1/2}) \\
& + u (2^{3/2} \hat{z} G^{-1/2} p_r^{1/4} q_r^{-1/4} + 2^{1/2} \hat{z} G^{-3/2} p_r^{-1/4} q_r^{-3/4} p_t) \\
& + (1 + \frac{1}{4} G^{-2} p_r^{-1} q_r^{-1} p_t^2 - G^{-1} p_r^{-1/2} q_r^{-1/2} p_t) = 0
\end{aligned} \tag{87}$$

Introducing the dimensionless variables

$$a = \frac{1}{2} G^{-1} p_r^{-1/2} q_r^{-1/2} p_t \quad \text{and} \quad b = \sqrt{2} \hat{z} G^{-1/2} p_r^{1/4} q_r^{-1/4} \tag{88}$$

we get

$$u^4 - 2bu^3 + (b^2 - 2a - 6)u^2 + 2b(1+a)u + (1-a)^2 = 0 \tag{89}$$

The root of this equation μ is a function of a and b and is given in the table of section 9. The values of the various elements r_{ij} of \underline{R} can now easily be calculated. With

$$\omega_o = G^{1/4} p_r^{1/8} q_r^{-1/8} \tag{90}$$

we get

$$r_{11} = q_r \lambda \omega_o^8 G^{-2} \tag{91.1}$$

$$r_{12} = q_r \lambda \omega_o^7 G^{-2} 2^{3/4} \mu^{1/2} \tag{91.2}$$

$$r_{13} = q_r \lambda \omega_o^6 G^{-2} 2^{1/2} \mu \tag{91.3}$$

$$r_{14} = q_r \lambda \omega_o^5 G^{-1} 2^{-3/4} \mu^{-1/2} (\mu^2 + b\mu + 1 - a) \tag{91.4}$$

$$r_{15} = q_r \lambda \omega_o^4 G^{-1} (b\mu + 1) \tag{91.5}$$

$$r_{22} = q_r \omega_o^7 G^{-2} 2^{3/4} \mu^{1/2} \tag{91.6}$$

$$r_{23} = q_r \omega_o^6 G^{-2} 2^{1/2} \mu \tag{91.7}$$

$$r_{24} = q_r \omega_o^5 G^{-1} 2^{-3/4} \mu^{-1/2} (\mu^2 + b\mu + 1 - a) \tag{91.8}$$

$$r_{25} = q_r \omega_o^4 G^{-1} (b\mu + 1) \tag{91.9}$$

$$r_{33} = q_r \omega_o^5 G^{-2} 2^{-3/4} \mu^{-1/2} (3\mu^2 - b\mu - 1 + a) \tag{91.10}$$

$$r_{34} = q_r \omega_o^4 G^{-1} (\mu^2 - a) \tag{91.11}$$

$$r_{35} = q_r \omega_o^3 G^{-1} 2^{-5/4} \mu^{-1/2} \{3b\mu^2 + (4-b^2)\mu + b(a-1)\} \tag{91.12}$$

$$r_{44} = q_r \omega_o^3 2^{-5/4} \mu^{-1/2} \{2\mu^3 - b\mu^2 + (b^2 - 2a - 2) + b(1-a)\} \tag{91.13}$$

$$r_{45} = q_r \omega_o^2 2^{-1/2} (b\mu^2 + 2\mu - ab) \tag{91.14}$$

$$r_{55} = q_r \omega_o 2^{-7/4} \mu^{-1/2} \{(2+3b^2)\mu^2 + b(6-b^2)\mu + (1-a)(2-b^2)\} \tag{91.15}$$

The feedback coefficients k_i can be calculated from (3), which for this case has the form

$$k_i = (r_{i5} - \hat{z} r_{i3}) q_r^{-1} \tag{92}$$

The difference between the αY -system and the βX -system is only in the elements a_{43} of \underline{A} and c_3 of \underline{C} . They are $-G$ and $+2$ for the αY -system and $+G$ and -2 for the βX -system. It can be verified directly from the equations (81), that the solution for the αY -system is obtained from the solution of the βX -system by changing the signs of the elements r_{ij} with $i \leq 3$ and $j \geq 4$ and then from (92) it follows that also the coefficients k_i with $i \leq 3$ change their sign.

The γ -system of section 9 is the same as that of section 7 and 5 and calculations of 11.2 are valid. The Z -system is a simple second order system. The matrices \underline{A} , \underline{C} , \underline{P} and \underline{Q} of this system as given in section 9 substituted in (3) results in

$$\begin{aligned}
r_{12}^2 &= p_t q_r \\
r_{12} r_{22} &= q_r r_{11} \\
r_{22}^2 &= q_r r_{12}
\end{aligned} \tag{93}$$

These equations can easily be solved:

$$r_{11} = 2^{1/2} p_t^{3/4} q_r^{1/4} \quad r_{12} = p_t^{1/2} q_r^{1/2} \quad r_{22} = 2^{1/2} p_t^{1/4} q_r^{3/4} \tag{94}$$

The feedback coefficients of fig. 9 are obtained from (94) and (3)

$$k_9 = p_t^{1/2} q_r^{-1/2} \quad k_{10} = 2^{1/2} p_t^{1/4} q_r^{-1/4} \tag{95}$$

REFERENCES

- 1 Offereins, R.P. Determination of ships orientation from accelerometer signals.
Agard Conference Proceedings no. 116 on Inertial Navigation Components and Systems,
paper 18, 1972.
- 2 Agard-LS-95 Strap-down inertial systems.
- 3 Agardograph 139 Theory and Applications of Kalman filtering.
- 4 Agard-LS-82 Practical aspects of Kalman filtering.
- 5 Arthur Gelb Applied Optimal Estimation.
(editor) M.I.T., 1974.
- 6 Veld, P. Program for attitude determination, real-time and simulation.
M.Sc. thesis (Dutch), 1974.
- 7 Tiernego, M.J.L. Analysis of a method for attitude determination with two accelerometers.
M.Sc. thesis (Dutch), 1976.
- 8 Luisman, J.H. Three dimensional attitude determination.
M.Sc. thesis (Dutch), 1977.

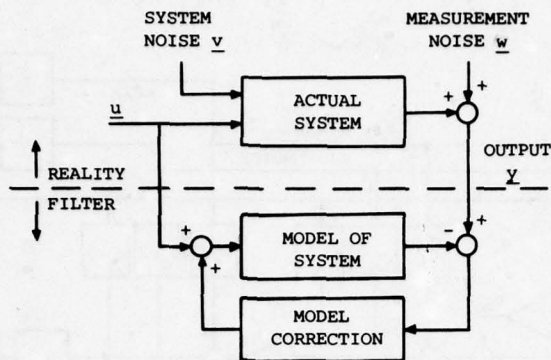


FIG. 1 PRINCIPLE OF KALMAN FILTER

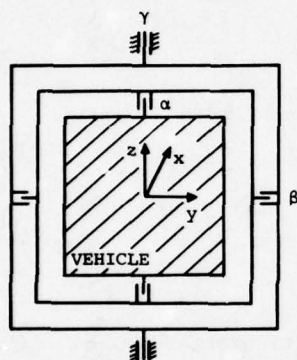


FIG. 2 EULER ANGLES

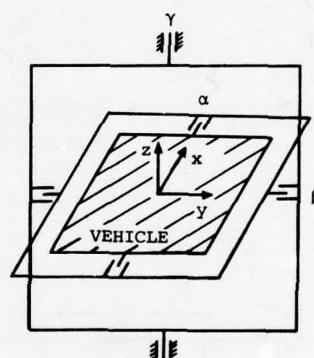


FIG. 3 GIMBAL ANGLES

v = SYSTEM NOISE
 w = MEASUREMENT NOISE
 \emptyset = INPUT SIGNALS

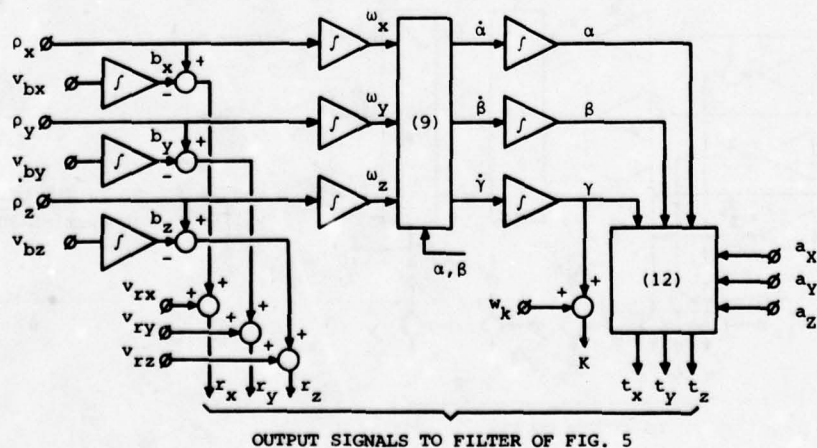


FIG. 4 MODEL FOR ATTITUDE GENERATION FROM ACCELERATION

$$\omega_o = G^{1/2} p_r^{1/4} q_a^{-1/4} \quad \omega_Y = p_r^{1/4} q_k^{-1/4} \quad \lambda = p_b^{1/2} p_r^{-1/2}$$

\emptyset = INPUT SIGNALS (FROM MODEL OF FIG. 4)

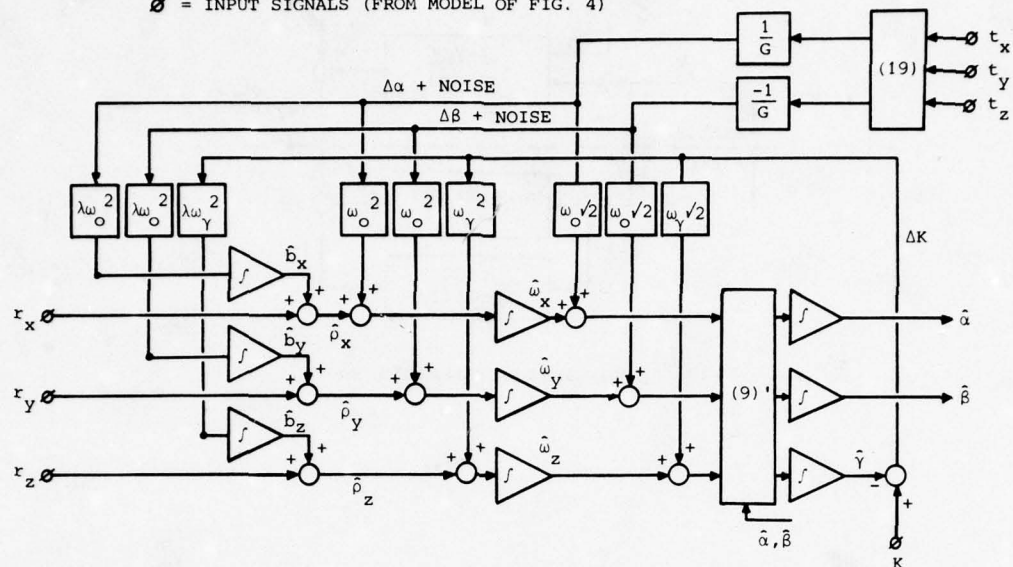


FIG. 5 FILTER FOR ATTITUDE DETERMINATION FROM ACCELERATION AND COURSE MEASUREMENTS

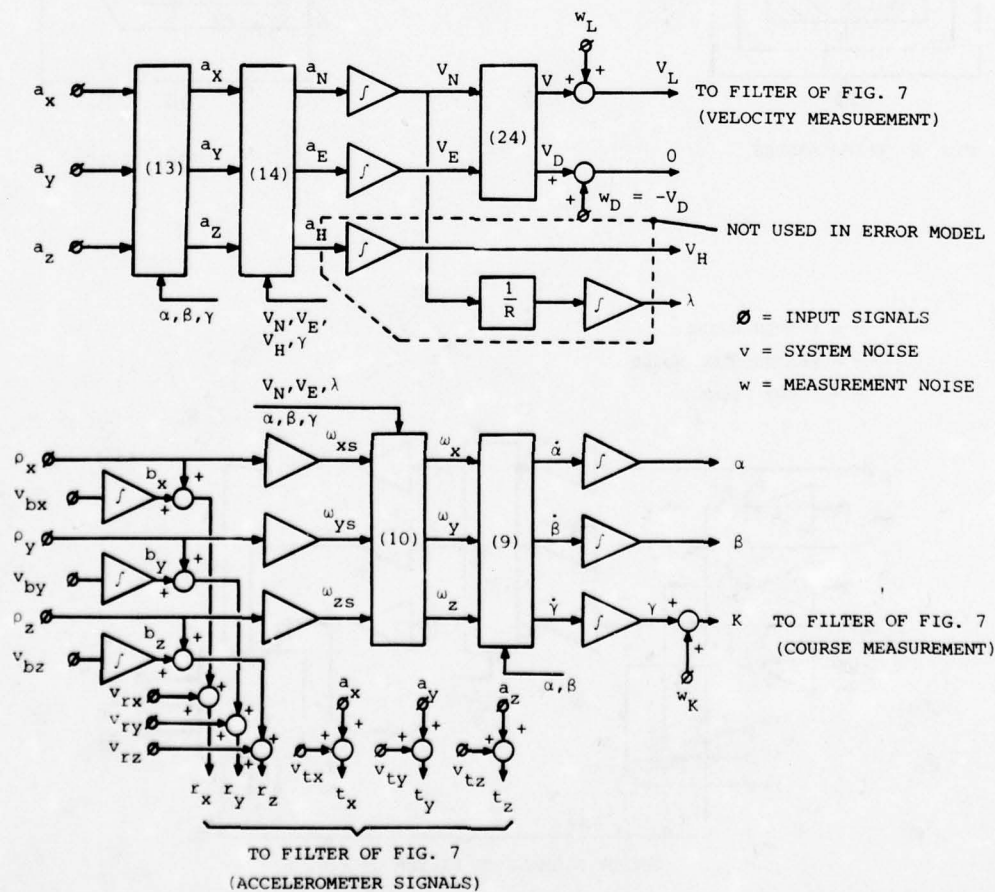


FIG. 6 MODEL FOR ATTITUDE AND VELOCITY GENERATION FROM ACCELERATION

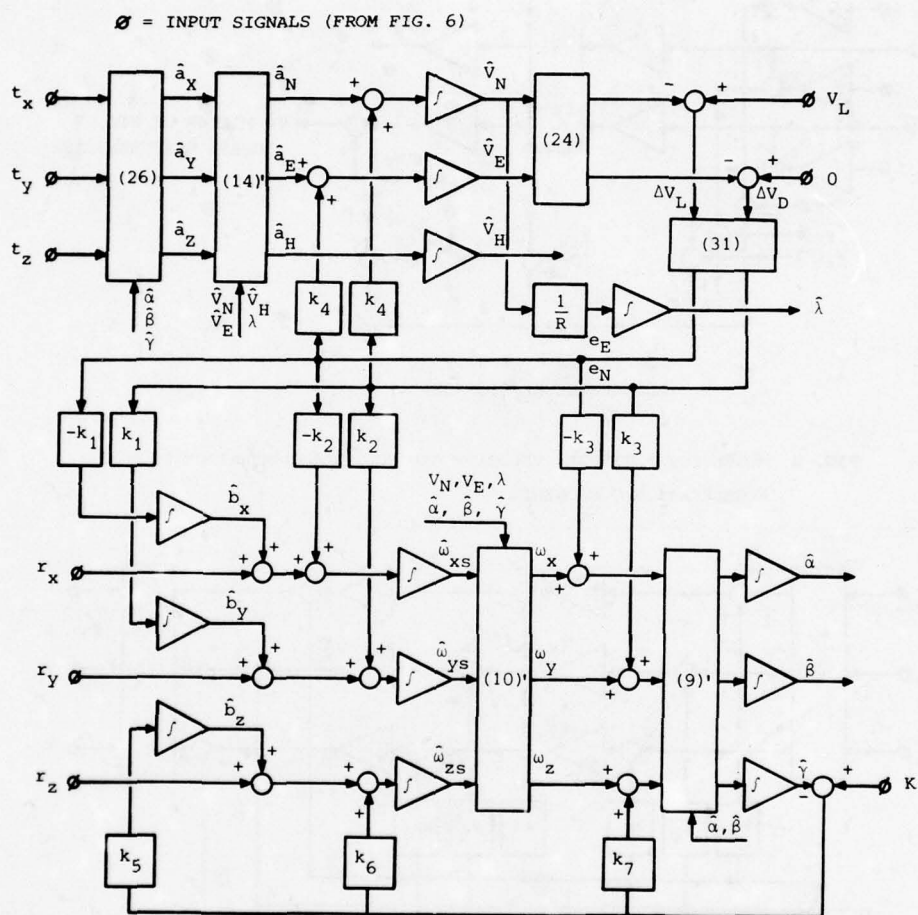


FIG. 7 FILTER FOR ATTITUDE AND VELOCITY DETERMINATION FROM
ACCELERATION, COURSE AND VELOCITY MEASUREMENTS

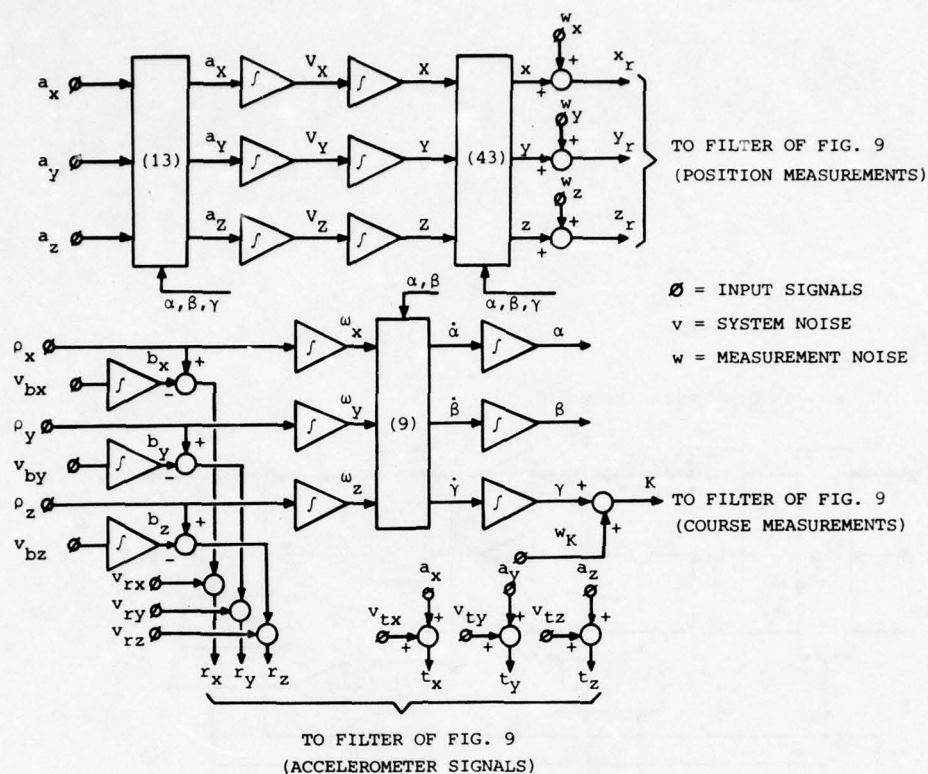
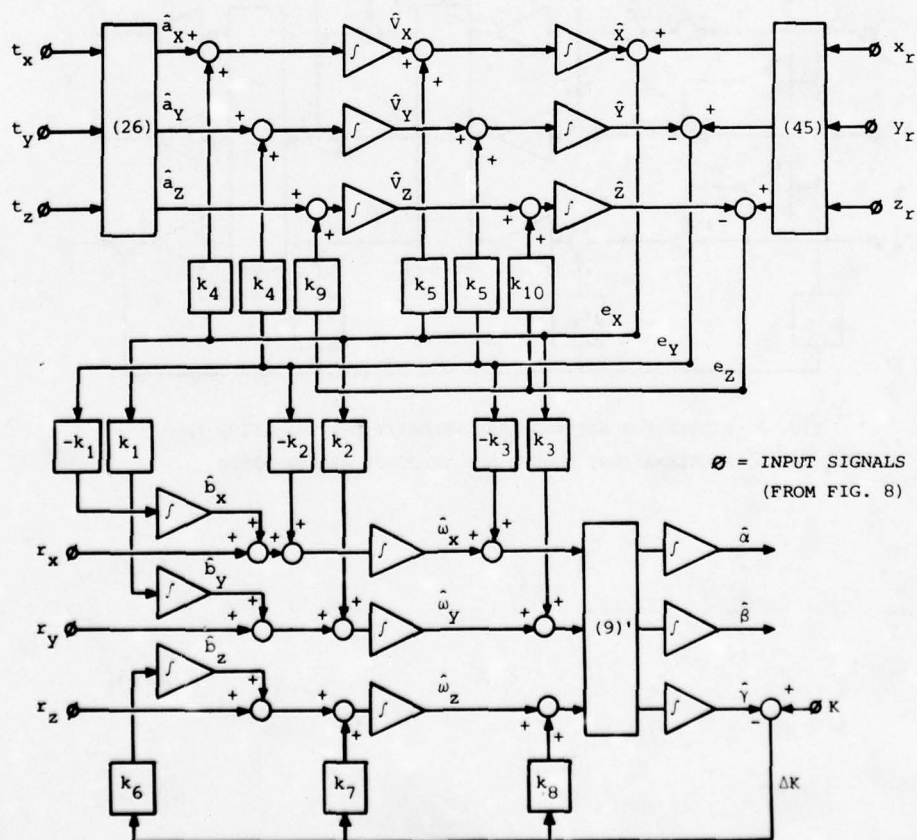


FIG. 8 MODEL FOR ATTITUDE, VELOCITY AND POSITION GENERATION FROM ACCELERATION COMPONENTS



DIGITAL SIGNAL PROCESSING TECHNIQUES IN A MONOPULSE TRACKING RADAR

U. Fazio, F. Ambrosioni, C. De Bonis

CONTRAVES ITALIANA
Via Tiburtina 965
00156 ROME
ITALY

SUMMARY

This paper describes a special-purpose Signal Processor, the development and realization of which is, at the present time, well underway. Since the Processor is highly flexible, it can be applied in a wide range of modern ground or ship-borne tracking radars, such as, for instance, Fire Control Radar for rockets or for conventional artillery (ground or anti-aircraft), Instrumentation Radar, Tracking Radar for Command to line-of-sight missile guidance. Herein are illustrated the three main functions of the processor, considered as being associated to a fully-coherent monopulse radar: signal enhancement based on an FFT (Fast Fourier Transform) algorithm, tracking errors computation and system logic control. Along with this, a detailed description is given, especially as regards the FFT part, of the practical realization of each block, based on a wide use of the most modern digital techniques and of microprogrammed controls in particular, that have enabled high flexibility and growth potential to be obtained. In the description, it can be seen how the design trade-off costs/performances has led to the predominant use of well-known techniques and widely used electronic circuitry, excluding custom components.

1. INTRODUCTION

Recent advances in radar techniques and associated technologies (in both the field of microwaves as well as that of electronic components, and digital components in particular) have led to the realization of improved tracking radars, as regards performances, reliability, and easy operation, whilst costs and overall dimensions have remained within the limits normally compatible with tactical use, even in the case of mobile ground or ship-borne systems. In addition, the ever wider use of minicomputers for the handling of output data from the radar itself has enabled greater and/or further advantages to be obtained, as compared with what has been possible until now, in the use of tracking radars for guiding and controlling, thereby expanding the field of application.

In this light, the design and construction of standardized functional blocks in the radar, the use of microprogrammed controllers and of distributed microprocessors enables "basic" tracking radars to be realized that, with modifications mainly to software and firmware, can be transformed into different custom-tailored systems: in other words, hardware today can be so flexible and multipurpose that it can be re-arranged in several different special-purpose radars covering, altogether, a great number of requirements in the field of guiding and controlling. In this respect, a particular example is given herein of a radar, presently under development, which was designed in this way: special attention will be given to the video digital signal processor functional block since it is the one that best illustrates the fundamental role played by modern digital techniques in problems common to all radars in the above category. Therefore, although we shall not be expressly talking about what is "before" (radar sensor) or "after" (tracker servo system) the signal processor, we shall mention here the major applications envisaged for the system, which, it will be noted, do not require exceptional accuracies or transmission power. Such applications are:

- a) Fire Control Radar, for ground or ship-borne applications, to be used in connection with conventional artillery;
- b) Tracking Radar, for the control of Surface-to-Air-Missiles using the Command to line-of-sight method, in both 'beacon' and 'skin' mode ;
- c) Instrumentation Radar, for detecting data of cooperative or non-cooperative targets, such as aircrafts, drones, artillery shells, etc. ;
- d) Tracking radar, for control and correction of conventional field artillery or rockets.

We will now take a brief look at the main features that such a tracking radar must have in the applications seen:

- it must be based on very narrow pulses, of the order of tenths of a microsecond at least, in order to have high range resolution;
- it must be able to eliminate ground- and weather-clutter with at least 40dB above the useful signal, in order to minimize the negative effects of clutter environment on tracking performances;
- it must have a transceiver and video signal processor capable of working with staggered Pulse Repetition Frequencies (PRF) in order to be able to receive optimal information from targets with a wide range of possible velocities (up to Mach 5);

- it must have high sensitivity and wide dynamics, in order to handle tracking range errors of the order of one metre, and angular errors of the order of one tenth of a milliradian, and for targets of very different sizes;
- it must be able to incorporate, both in the transceiver and in the video signal processor, a large number of ECCM (Electronic Counter Counter Measures) devices, to be employed quite elastically according to the application and to the foreseen ECM threat;
- the number and level of automatisms must be such as to be easily handled by the human operator, even in the case of quite complex or sophisticated procedures;
- it must be able to carry out smoothing, filtering and data compression operations, in order to link up efficiently with an 'external, intelligent' system (host computer, data collection system, etc.), as well as have a simple but powerful interface;
- it must be very flexible and powerful, so that it can be easily modified or expanded upon, especially in the field of adaptive digital techniques both as regards environment (f.i. clutter locking) and as regards targets (f.i. velocity tracking, multipath effect reduction).

In the case we are illustrating, these requirements have been taken into account, as regards the radar with the adoption of a monopulse tracking system, of a fully coherent transceiver and of an antenna-transmitter system able to guarantee the required range performance.

As far as the signal processor is concerned, the key features, that will be illustrated in detail later, are as follows:

- clutter elimination using a doppler filters bank, chosen on account of its well-known features of good signal matching and of intrinsic flexibility and growth capability: the filters bank is a processor that uses the algorithm [1] Fast Fourier Transform (FFT); a weighting operation, in the time domain, of input samples, with Dolph-Chebyshev [2] type window, reduces the filters frequency sidelobes, and by simply acting on its weight coefficients, enables the clutter cancellation ratio, desired for each application, to be obtained;
- vectorial calculation of the tracking errors and handling of the processes associated thereto by means of a high speed bit-slice microcomputer, chosen to achieve major potentiality as compared with conventional special purpose hardware;
- control based on microprogrammed logic, that handles logic, timing and interfacing among the various units and between the system and the outside; this has enabled study costs and the amount of hardware to be kept down, even though timing sequences and complex controls have been realized for a large variety of radar operative modes and PRF's.

2. SIGNAL PROCESSING ARCHITECTURE

We shall now describe the Signal Processing unit applied in the tracking radars as illustrated above: this unit receives the input video signals from the radar receiver Phase Sensitive Detectors circuits; such signals are of the I-Q type, that is, they come from a phase-quadrature type demodulation [3]. The Signal Processing hardware architecture (ref. block diagram, Fig. 2.1) considered here is made up of:

- "Data Acquisition System": unit that converts the radar video signal from analog to digital and reorders data for the
- "FFT Processor": special purpose processor that carries out FFT algorithm on radar data and supplies the results to an;
- "FFT Post Processor": special purpose processor that carries out processing to detect the presence and position of any targets (target acquisition), extracts the most significant data of the tracked targets and generates signals for video display of the radar signal (target tracking). Data pertaining to the detection of any targets or to the tracking of already acquired targets are sent to the
- "Radar Data Processor": high speed general purpose microcomputer that, by carrying out mathematical/logic operations on such data, supplies the outside with information on the target, controls the range axis and modifies the parameters of the various radar units in order to optimize acquisition and/or tracking. All the radar units (transmitter, receiver, etc.) are connected to each other and to the Signal Processing by a digital bus controlled by an 8080 type microcomputer. Such unit, called
- "Logic Data Processor" handles medium speed information exchange among the various radar units, carries out supervision of the system status and controls the various test points and execution of the BITE (Built-in-test equipment) programs.

The following paragraphs briefly describe the various units that make up the Signal Processing System, with particular attention to the "FFT Processor", the general characteristics of which will be illustrated, together with its signal filtering behaviour, problems connected with the type of realization chosen, the prototype main characteristics and the results of the tests carried out on it.

In addition, we shall look at the advantages and potentiality of microprogramming in the development of the Radar Data Processor unit.

3. DATA ACQUISITION SYSTEM

In this sub-unit, the quadrature and phase components of the bipolar radar signals are sampled in range-gate order with high speed S/H's and are digitalized at 8 bits with a max. rate of 100 ns per conversion; as regards the tracking error channels, since more accuracy is required, conversion is carried out with 12 bits at lower speed. Once the data have been sampled and converted, they are sent in FIFO (First-In-First-Out) circuits that act as interface with the FFT Processor block.

The converters are of a hybrid (medium speed-high precision) and monolithic (high speed) type, the latter being realized with triple diffusion (3D) bipolar technologies.

The timer, under the control of the Radar Data Processor, governs the range axis, generating trigger pulses for the S/H's, for the A/D, for the FIFO and for the other system units, and on account of the high speeds involved, is partly realized with ECL logic suitably interfaced with TTL logic.

4. FFT PROCESSOR

4.1 FFT Algorithm

The DFT (Discrete Fourier Transform) is an efficient means for filtering noise immersed signals and can therefore be applied for radar signal processing. The FFT algorithm enables the number of operations needed for DFT computation to be reduced quite drastically. As is known [4] DFT is a digital operator that transforms a time sequence of signal samples into a sequence of an equal number of samples that, once suitable sampling conditions have been met, represent the signal frequency spectrum samples. The DFT of a sequence of N complex samples, equally spaced at T seconds, is defined as:

$$X(k) = \sum_{n=0}^{N-1} x_n W^{nk} \quad k=0,1,\dots,N-1 \quad (4.1.1)$$

where:

$$W = e^{-j2\pi/N} \quad ; \quad j = (-1)^{1/2} \quad (4.1.2)$$

The spectral samples $X(k)$ are frequency spaced at $1/NT$ Hz. Direct computation of a DFT requires N^2 complex multiplications and $N(N-1)$ complex additions. When N is a power of 2 the FFT algorithm radix 2 requires a number of elementary operations (1 complex multiplication + 2 complex additions) equal to:

$$P(N) = N/2 \log_2 N \quad (4.1.3)$$

Such elementary operation is normally called 'butterfly' and it is shown graphically in Fig. 4.1 (where A, B, W, A', B' are complex numbers) for the DIT (Decimation-in-Time) version that is the one we have adopted. As is known, an 'in place' implementation is possible for such version, that is, with a number of memory cells equal to N , without the need for further memory locations for the storage of intermediate results. Fig. 4.2 shows, as an example, the flow diagram of the FFT radix 2 for $N = 8$, using the above DIT technique.

4.2 Behaviour of FFT Processor and its application in doppler filtering

It is known that the FFT can be considered equivalent to a parallel bank of passband linear filters that have the same shape but with frequency shifted transfer function. It is shown pictorially in Fig. 4.3.

It is due to this property that the FFT algorithm is interesting in many radar signal processing applications and becomes preferable to traditional MTI filtering realized with analog or digital high-pass filters. The FFT's intrinsic flexibility enables mobile clutter, in particular, to be eliminated as long as this has a different doppler frequency from that of the target. Figs. 4.4 and 4.5 show this situation analysed with a single band MTI filter and in comparison with the FFT: with the former, clutter that has entered in band cannot be eliminated; with the FFT the filter or filters that contain it have to be merely excluded.

Here below are briefly summarized the main advantages of the FFT as compared with conventional filter in doppler processing:

- improvement of the signal/noise ratio due to coherent integration;
- improvement of the signal/interference ratio for wide band interference:
 - only the disturbance that falls in the doppler filter matched with the target speed can affect the signal;
- major flexibility of the filter shape;
- possibility of carrying out velocity tracking;
- own motion compensation for radar systems mounted on mobile units.

4.3 Choice of the type of realization

Solutions along the lines of the Array Processor (that has $N/2 \log_2 N$ elementary units) or partly along the lines of the Array-iterative Processor (that has $N/2$ arithmetic units) or Pipeline Processor (that requires $\log_2 N$ arithmetic units cascade connected by multiplexer) have not been taken into consideration since they were deemed to be too complex and costly. Rather, a serial solution has been adopted that has a single arithmetic unit that carries out in sequence all the FFT elementary operations (butterfly). The type of elementary operation is $A+WB$ where A and B are data (complex) and W is a weight (complex). The real and imaginary components of the complex numbers for the DIT butterfly are expressed as:

$$\begin{aligned} A'_R &= A_R + B_R W_R - B_I W_I \\ B'_R &= A_R - B_R W_R + B_I W_I \\ A'_I &= A_I + B_R W_I + B_I W_R \\ B'_I &= A_I - B_R W_I - B_I W_R \end{aligned} \quad (4.3.1)$$

A direct approach, completely along the lines of the DIT butterfly realization requires (see Fig. 4.6):

- 4 real multipliers
- 6 real adders
- 4 RAM (Random-Access-Memory) for data
- 2 PROM (Programmable-Read-Only-Memory) for the weights.

The butterfly processor in this case would be very speedy but very bulky and it would be difficult to insert it in a single mean-size electronic board on account of the high number of components and Input-Output signals.

Instead, we have used a microprogrammed serial approach: in fact, the butterfly is carried out using a sequence of steps, each of which is controlled by a microprogram memory.

In other words, the controls needed (e.g. read-write commands, enable clock, select multiplexer, etc.) in the various processor points (memories, arithmetic logic unit, latches, etc.) are not generated by combinations of suitably connected gates and flip-flops but are obtained from a memory in which they were previously written in a permanent way and their read-out addresses are generated sequentially [5]. In this case, the following are necessary for realization (see Fig. 4.7):

- 1 real multiplier
- 1 real adder
- 1 RAM for data
- 1 PROM for weights
- 1 PROM for controls (micro-program memory)
- 1 sequencer.

It is clear that with this philosophy, the major effort is that of obtaining sequences of microinstructions as short as possible so as to minimize the time required for carrying out the operation so as to be compatible with the hardware that we want to use.

4.4 FFT Processor Description

The sampled and converted videos taken from the output FIFO of the Data Acquisition System are memorized in bit-reversed order in an FFT block memory governed by an acquisition controller. The processor incorporates a Dolph-Chebyshev smoothing window that reduces the amplitude of the sidelobes in the FFT transfer function to the desired level. The window is applied in the time domain by means of multiplication during loading of data into the acquisition memory: this technique was chosen rather than the equivalent frequency windowing, since it is simpler and more flexible. Once the data have been acquired in the memory, the latter is handled by a processing controller while, at the same time, another memory stores the new data; once processing has been completed, the two memories are exchanged so that the process is continuous and without information loss.

To reach the velocity required in the FFT processor, a simple and flexible parallel architecture has been used: the basic idea arises from the observation that the signal filtering is identical in all the range gates, so that, while the memory and calculation units must be parallel, a single control unit is sufficient; therefore, the FFT processor is made up of several identical memory and calculation units handled simultaneously by a common control unit. In our application, there are 8 processing units working simultaneously. The concept is illustrated in Figs. 4.8 and 4.9. In the latter, which shows the FFT processor block diagram, it should be noted that there is a BITE block that enables automatic testing of the sub-units to be carried out in order to obtain rapid replacement of faulty circuit boards.

Through the use of commercially available Large Scale Integration (L.S.I.) components (multipliers, accumulators, RAM, PROM, etc.) it has been possible to insert the calculation unit that implements the butterfly and the 2 memory banks (each having 1K words capacity of 12 bits) in a single board measuring 273 x 151 mm.: the two memory banks are sufficient for memorizing the quadrature and phase components of a number of range-gates equal to $512/N$ (where N is the number of signal samples).

Table 4.1 below summarizes the more salient characteristics of the FFT Processor; as regards flexibility, it should be noted that:

- it enables processing of sequences of length N variable from 8 to 64 for a corresponding maximum number of range-gates from 512 to 64;
- it enables the realization of scaled-down version (i.e. with fewer range-gates and/or N samples) simply by excluding some of the memory and calculation cards.

Type of algorithm	FFT, Radix 2, Decimation in Time, In Place
Type of arithmetic	12 bits for weights and for data, fixed point (24 bits for complex words)
Butterfly time	Butterfly cycle: $T_B = 3.2 \mu s$
FFT processing time	$T_{FFT} = \frac{N}{2} (\log_2 N) \cdot T_B$; $N = 32 \rightarrow T_{FFT} = 256 \mu s$
Number of printed boards (multilayer)	8 calculus and memory (identical) 1 time weighting and data acquisition controller 1 processing controller 1 B.I.T.E. (Built In Test Equipment)
Dissipation	Average 12 W per board
Flexibility	Possibility of working with $N = 8, 16, 32, 64$ for a maximum of 512, 256, 128, 64 range gates respectively. Possibility of working with a reduced number of calculus boards without changing the hardware.

Table 4.1: FFT processor main characteristics

Fig. 4.10 shows a photograph of the memory and calculation unit realized by us. The following digital components were used: SSI (Small-Scale-Integration), MSI (Medium-Scale-Integration) of the TTL Schottky or Low Power Schottky type and various LSI (Large-Scale-Integration) circuits, both bipolar (multipliers, PROM, FIFO, sequencer) and N-MOS (1K x 4 static RAM's).

4.5 Laboratory test results

The FFT Processor unit has been completely realized as a bread-board; it was checked by using the BITE itself and above all by using logic state analyzers: the aim was to check the hardware and to carry out functional tests. The tests carried out at the nominal clock frequency have closely agreed with theoretic results: Fig. 4.11 shows the behaviour of the amplitude response obtained through experiments.

5. FFT POST PROCESSOR

The main blocks are made up of:

- a modulus extractor that calculates, for each range-gate, the modulus relative to the N spectral samples, supplied by the FFT block in the form of I-Q components;
- a threshold detector realized with a range cell averaging C.F.A.R. (Constant False Alarm Rate) circuit [6] operating on each spectral sample. The CFAR processing is carried out according to a mask that excludes the clutter frequencies, thereby carrying out the MTI function.

Further circuit blocks are used for memorizing tracking error signal spectral components and for surveying statistical data on the noise that is present. In addition, there are interfaces for two possible indicators, the PPI which receives the two level output of the CFAR detector and the A-R Scope which receives the frequency averaged outputs of the

modulus extractor.

All these functions are put into effect by an assembly of 8 printed circuit boards, 273 x 151 mm, which include the control units and the BITE unit. The components used are of the same type as those employed in the FFT Processor.

6. RADAR DATA PROCESSOR

6.1 Tasks and functions

All the signals and main data that have been previously processed or prepared by other units of the Signal Processor pass through this unit for final processing, including any necessary arithmetic or logic correlation with each other.

In particular, whatever the specific application (see paragraph 1), operations of the following kind are carried out in the Radar Data Processor:

- handling of the range axis during the various radar operative modes (acquisition, tracking, etc.);
- vector calculations of the tracking angular errors, starting with the signals supplied by the FFT Post Processor;
- adaptive determination of the radar receiver gains and, in particular, of the Automatic Gain Control signals;
- handling data flow with the outside.

In addition, depending on the particular application, the processor may have one or more additional auxiliary functions, such as, for example:

- multiple time-around radar tracking capability;
- choice of the optimal Pulse Repetition Frequency sets;
- determination of the anticlutter filtering mask (to be applied to the FFT Post Processor);
- anomalous tracking status detection (e.g.: jamming, multipath, etc.) and any appropriate countermeasures.

Since the functions carried out are numerous and varied, it can easily be understood how the choice, as regards processor implementation, has had to be that of a solution based on a general-purpose microcomputer, with enough speed to cover the radar requirements described above.

6.2 Description of the Radar Data Processor Implementation

As has already been stated, such unit is a general-purpose microprogrammable processor (microcomputer), designed with the bit slice 2900 series of low power Schottky LSI bipolar integrated circuits, originally introduced by Advanced Micro Devices company and now produced by many other companies, on account of its success on the market. It has the following main characteristics:

- 16 bit fully pipelined architecture;
- 64 K words memory, directly addressable, with 10 addressing modes;
- 8 levels of priority interrupt;
- microprogram memory of 512 words each of 72 bits.

To give an idea of the machine's execution speed, it can carry out a register-to-register addition in about 300 nanoseconds or a memory-to-register addition in about 900 nanoseconds, throughout the whole military temperature range.

The heart of this microcomputer is the Am 2903 circuit, a 4-bit expandable bipolar microprocessor slice[7]. Four slices are connected to each other to form the 16-bit structure of the ALU (Arithmetic Logic Unit) and of the 16 connected general purpose registers. These four chips provide for the major data-path requirements with a minimum number of external control circuits. The most important of these support circuits is the control unit. It determines the processor instructions set power and the CPU flexibility.

A control unit has been built, completely microprogrammed with the Am 2910 microprogram controller circuit connected to 9 bipolar 512 x 8 PROM's.

As has already been stated, the result is a control memory of 512 words of 72 bits that enables single-phase micro-instruction with high operation parallelism that increases speed and enables microprograms to be carried out with flexibility and simplicity.

Microinstruction pipelining, instruction-stream pipelining, auxiliary logics for address computation and overlapping of decoding and of carrying out an instruction have been the techniques used to efficiently match the memory and the ALU with the control unit.[8]

The Central Processing Unit is contained in a printed-circuit board 273 x 151 mm. ; the high speed memory for the program and the data is on two printed boards. A further two boards contain the high speed serial interfaces for Radar Signal Processing External Computer and the parallel interfaces for the other Signal Processing sub-units.

7. LOGIC DATA PROCESSOR

This, too, is a general purpose microcomputer designed, however, with the LSI N-MOS circuits of the 8080 family originally introduced by Intel and now supplied by many other manufacturers.

The CPU printed board includes a Multimode DMA (Direct Memory Access) Controller with memory-to-memory data transfer capability. This characteristic speeds up data interchanges in a 12-bit control/data bus (8 bit: data/address, 4 bit: controls). This bus joins together all the radar units (Antenna, Transmitter, Low Power RF, Signal Processing, Control Panel) and allows exchange of data relative to the status of the individual units and to the system operative modes.

The CPU, the memory and the interfaces are contained in 8 printed boards, 170 x 150 mm.

8. CONCLUSIONS

This paper has described a Digital Signal Processing Unit which is quite flexible and therefore could be applied in a wide range of monopulse tracking radars. The unit is characterized by wide use of advanced digital techniques, such as microprogramming and time overlapping of memory/arithmetic operations.

The various sub-units, made up of microprogrammed blocks, some special-purpose others general-purpose, have been described from a functional and structural point of view, pointing out the advantages that the above mentioned techniques have yielded. As regards the sub-unit that implements the FFT algorithm, the test results obtained on the bread board (See Fig. 4.11) have been given.

REFERENCES

1. T.Bailly: The Fast Fourier Transform and its application to radar signal processing.
International Conference on Radar - Paris 4/8 Dec. 1978.
2. H.R. Ward: Properties of Dolph-Chebyshev weighting functions.
IEEE Trans. on Aerosp. and Elect. syst. - Vol. AES-9 - Sept. 1973
3. G.L. Turin: An introduction to Digital Matched Filters.
Proc. IEEE - Vol. 64, N. 7, July 1976.
4. W.T. Cochran et al: What is the Fast Fourier Transform?
IEEE Trans. on Aud. and Elect. - Vol. AU-15 - June 1967.
5. T.M. Hedges: Replacing hardwired logics with microcode.
Electronics, Nov. 9 - 1978.
6. H.M. Finn, R.S. Johnson: Adaptive detection mode with threshold etc.
RCA Review - Sept. 1968.
7. Advanced Micro Devices, Inc.: The Am 2900 family data book.
1978 - Sunnyvale - CA.
8. G.F. Muething jr.: Designing the maximum performance into bit-slice minicomputers.
Electronics, Sept. 30 - 1976.

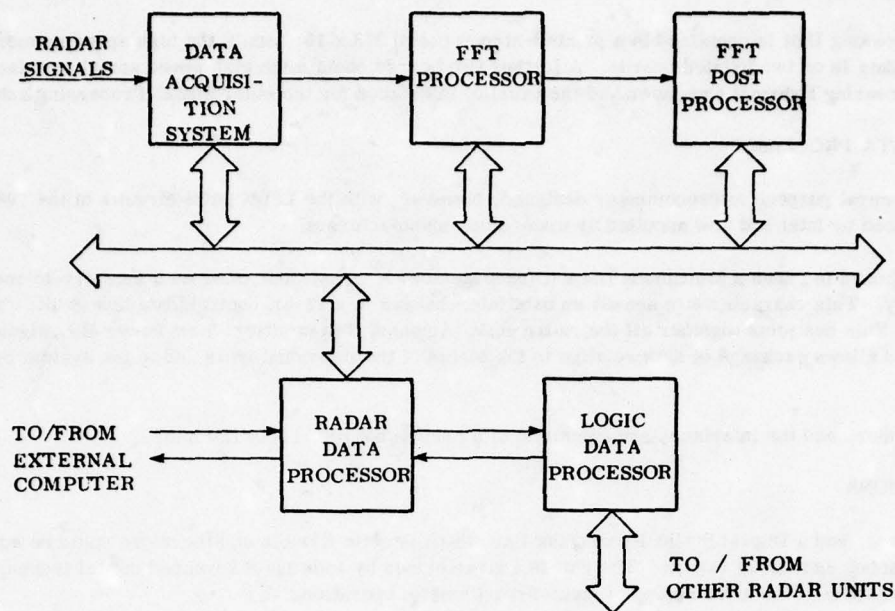


Figure 2.1 Signal Processing Unit Block Diagram

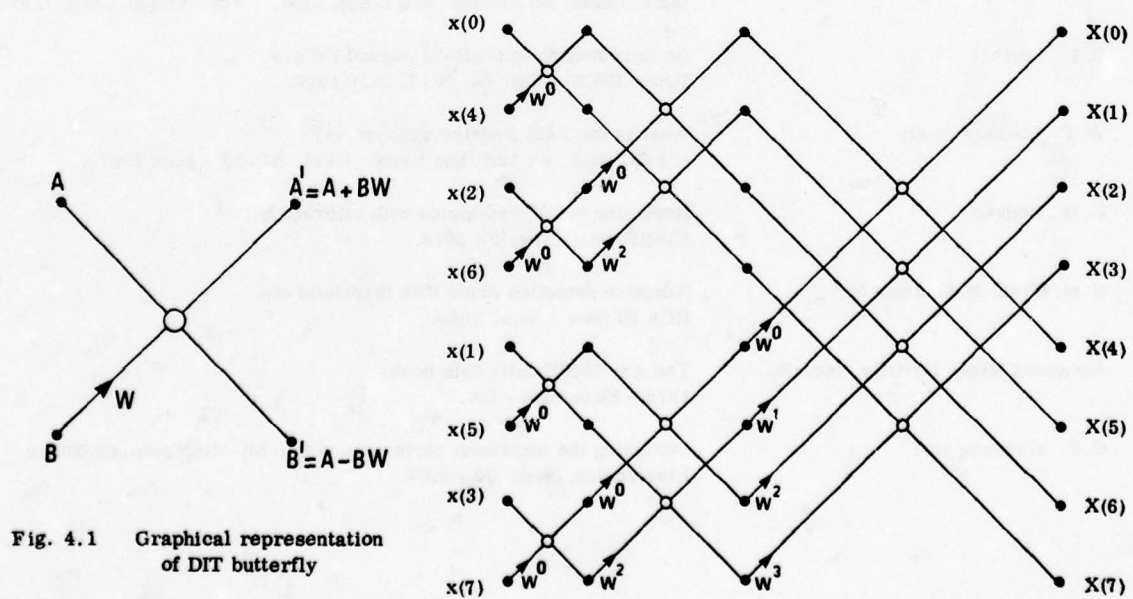


Fig. 4.1 Graphical representation of DIT butterfly

Fig. 4.2 FFT graph for N = 8 (DIT version)

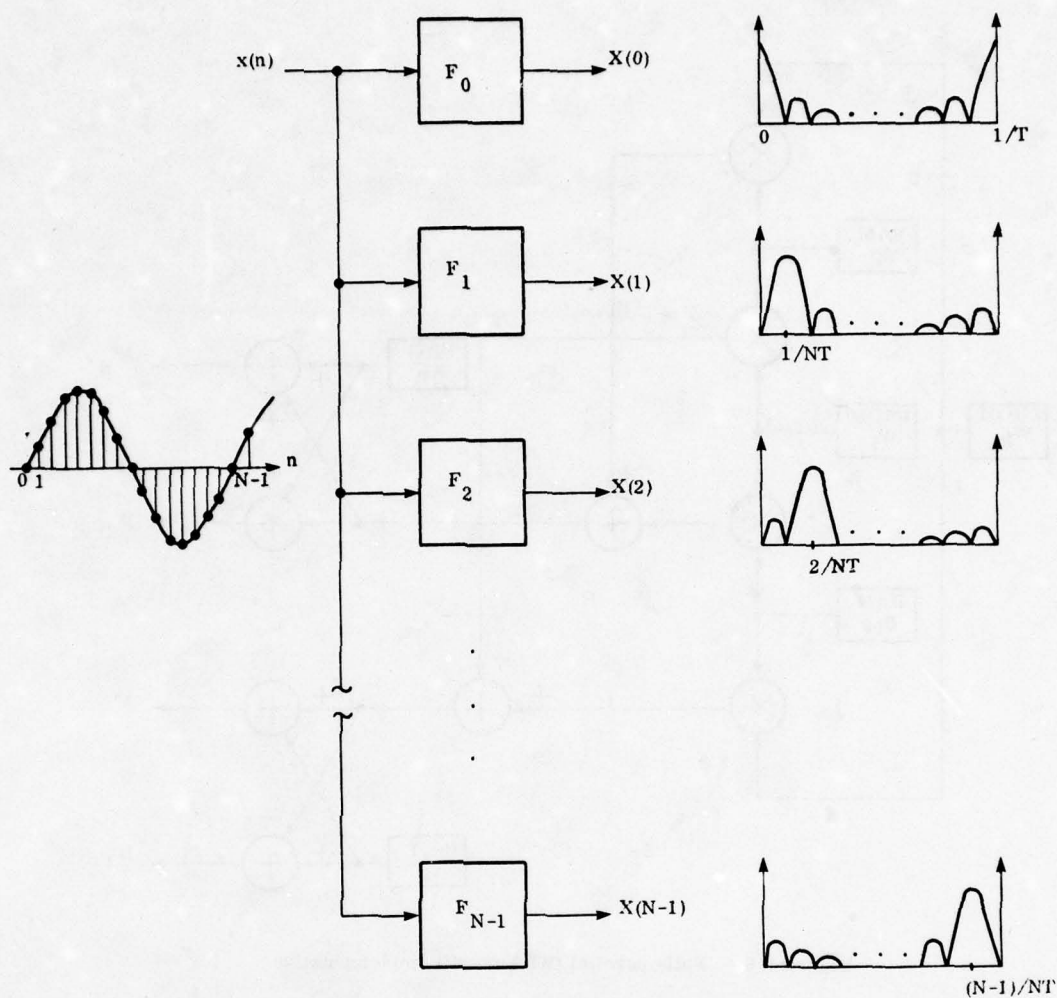


Figure 4.3 FFT Filter Bank Frequency Response

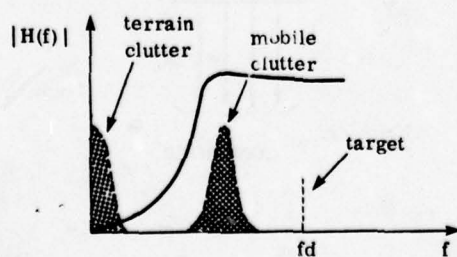


Figure 4.4 Mobile clutter not cancelled by an MTI filter

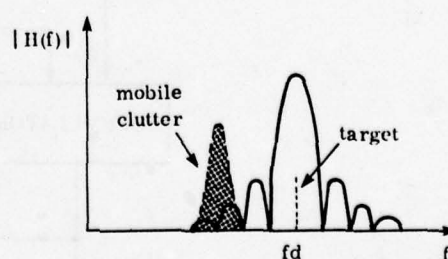


Figure 4.5 Mobile clutter cancelled by FFT

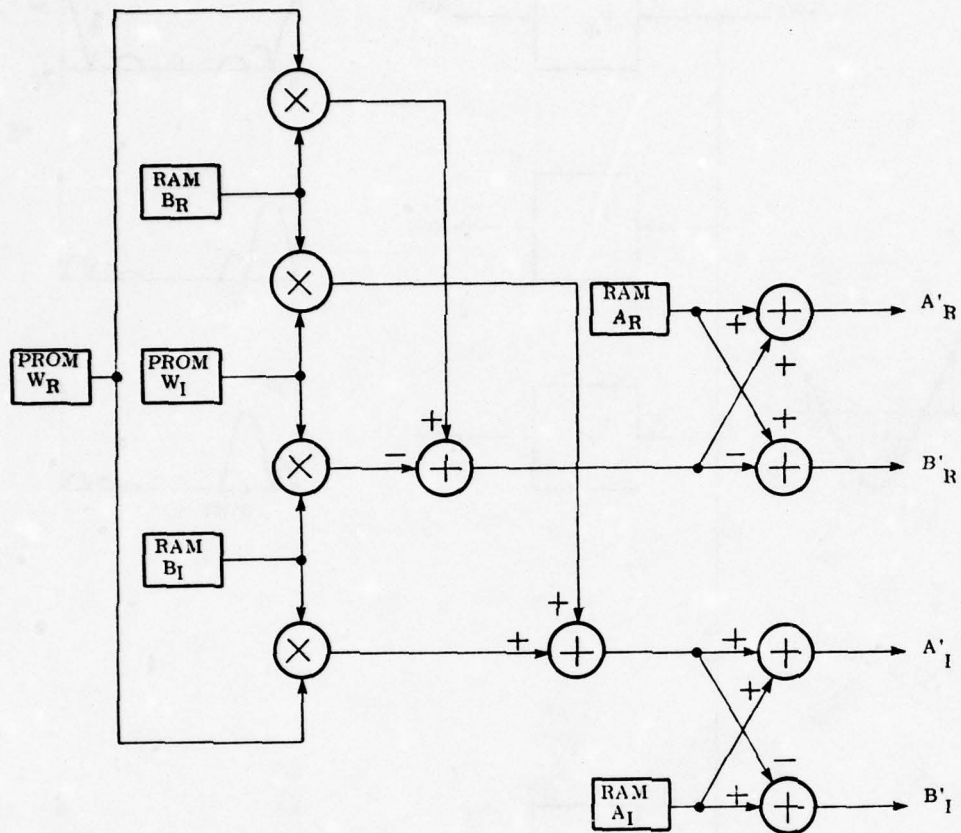


Figure 4.6 Fully parallel DIT butterfly implementation

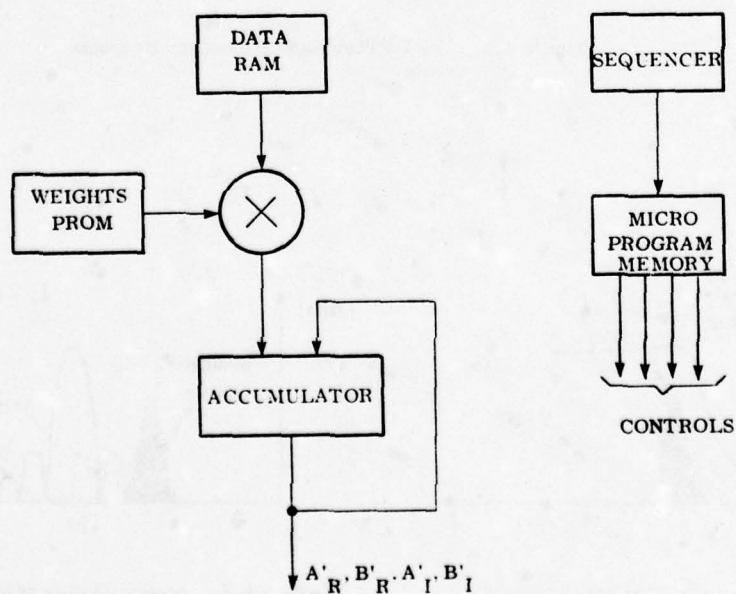


Figure 4.7 Microprogrammed DIT butterfly implementation

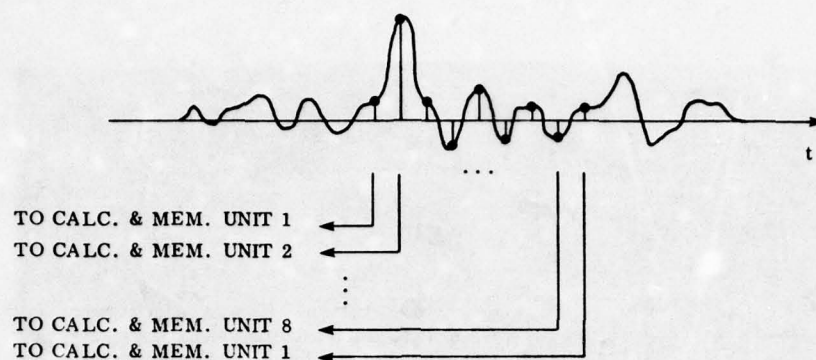


Figure 4.8 Range gate distribution in the various calculus and memory boards

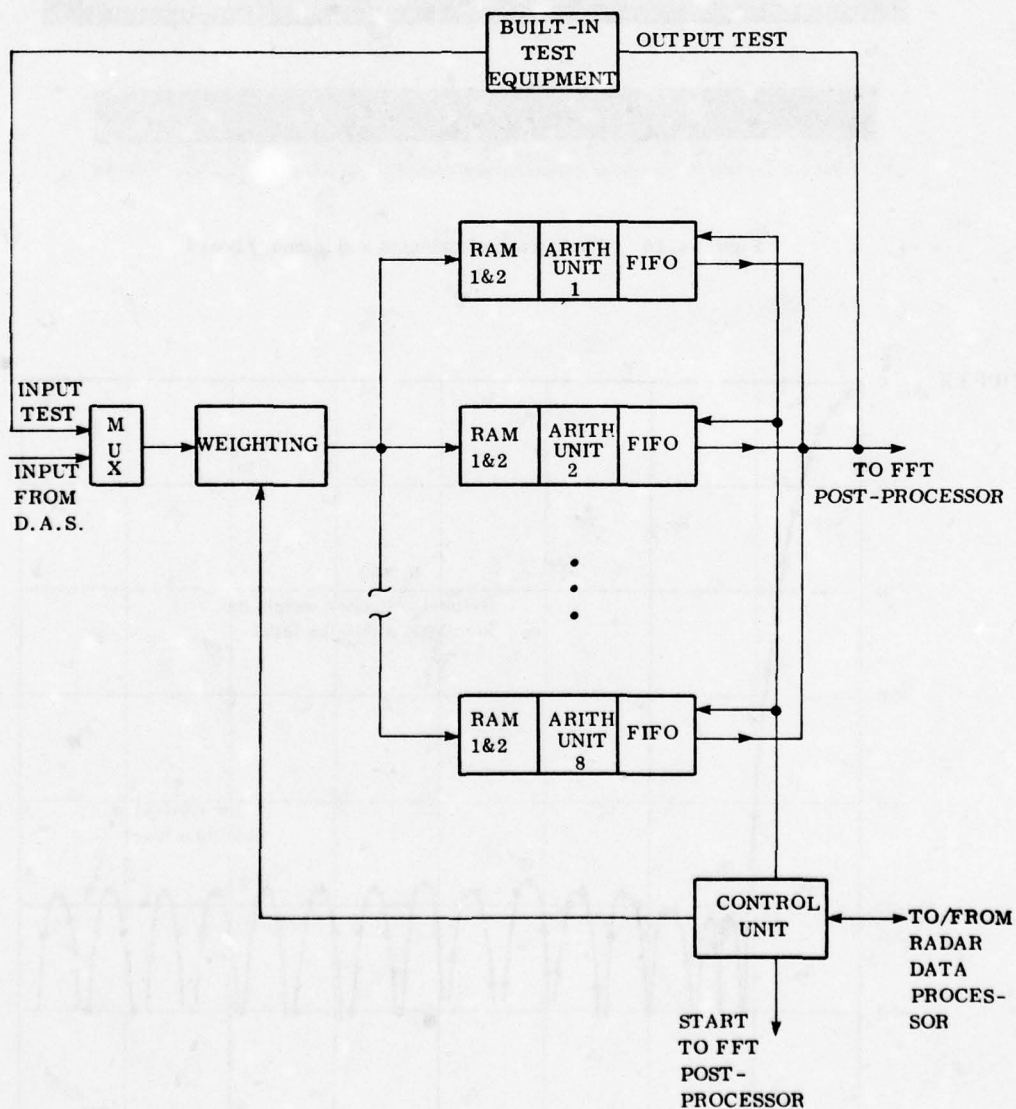


Figure 4.9 FFT Processor Block Diagram

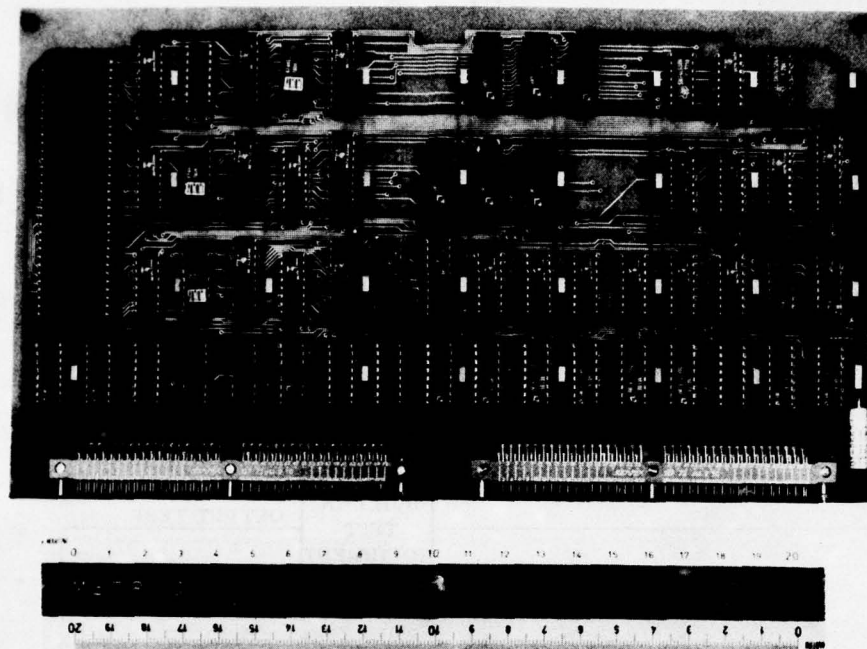


Figure 4.10 Photograph of calculus and memory board

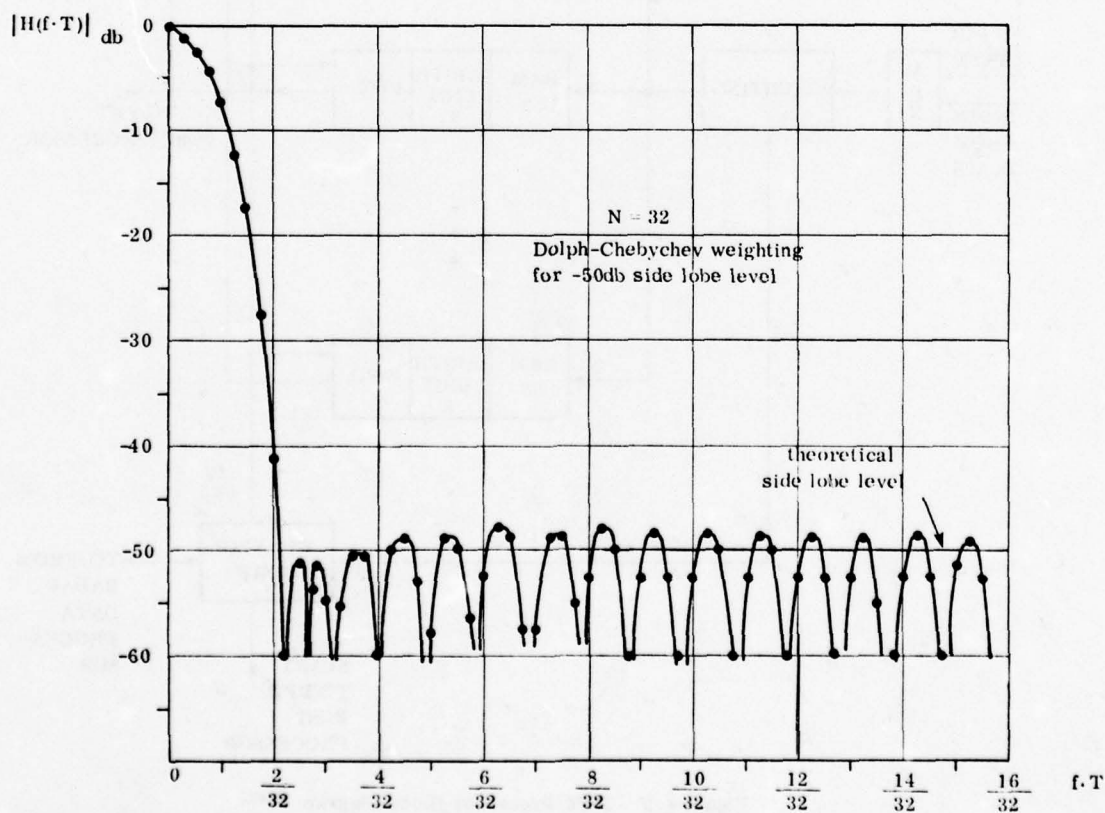


Figure 4.11 FFT amplitude response obtained experimentally

AN ASSESSMENT OF AND APPROACH TO THE VALIDATION OF DIGITAL FLIGHT CONTROL SYSTEMS

by
D. B. Mulcare and W. G. Ness
Lockheed-Georgia Company
Marietta, Georgia 30063, US

SUMMARY

Digital flight control systems (DFCS) present unique and increasingly demanding validation problems which challenge the capability of existing assurance methods and practices. Rooted largely in the nature and scope of the flight software, these problems are becoming more urgent owing to the trends toward more flight-critical functions and more extensive system integration. Both of these trends tend to cause increased complexity, which already is a matter of significant concern. Future systems, moreover, whose requirements will reflect substantial departures from current systems, add another dimension to system validation because of difficulties of ill-defined user needs.

An expanded role of system engineering, oriented toward the purposeful incorporation and confirmation of quality, is viewed as vital to the timely and economical deployment of future DFCS. An integrated methodology which judiciously blends flight control engineering experience and software engineering initiatives is therefore advocated. This methodology depends heavily on simulation, complemented by definitive analyses, from the conceptual phase onward.

INTRODUCTION

Cost and performance benefits, actual or projected, have motivated the rather widespread transition to digital mechanization of flight control systems (FCS). Furthermore, rapid advances in digital implementation technology and expanding system requirements render analog mechanization of many system functions impractical and in some cases infeasible.

An increasingly intense operational environment and crucial aircraft performance or maintenance related functions, as summarized in Table I, are expected to result in quite substantially revised system architectures and flight crew interfaces. These new architectures will likely be more airplane dependent, as contrasted with current autopilot functions, which are merely adapted to particular airplane configurations.

This is projected largely on variations among active control technology (ACT) functions, which are not simply add-on systems, but which must be tailored to very specific airframe characteristics. System functions and criticality will tend to vary appreciably among airplane configurations, and hence a corresponding variation among system architectures is anticipated. Computer subsystem architectures and system interfacing options, moreover, will tend to facilitate variations among system configurations. Other factors which indicate the need for more powerful and flexible validation methods are presented in Table II.

Background

It remains rather prevalent to equate verification and validation (V&V) with testing. Also, the tendency persists to view V&V activities as strictly after-the-fact evaluations of completed software products. In both of these instances, such oversimplifications can significantly limit the capacity to realize a quality DFCS product within a reasonable expenditure of resources.

Practical limitations on testing have long been recognized in both military and civil sectors of aviation. For example, MIL-F-9490D (Ref. 1) states that "To the maximum extent feasible, compliance with quantitative requirements of the FCS specification shall be demonstrated by tests." In a more explicit instance, FAA Advisory Circular AC-20-57A (Ref. 2) discusses the generation of a statistically significant number of autoland simulations, which would be unachievable in flight testing. Additionally, it addresses the correlation of the simulation data with flight testing.

A similar acknowledgement of the limitations of testing is applicable in the case of DFCS software. Basically, the problem lies in the infeasibility of exhaustive testing, which becomes even more formidable as the effects of various hardware faults are considered as well. While some examples of contemporary DFCS software exhibit only a modest degree of complexity, this will not generally apply to future DFCS mechanizations. Thus, the software engineering techniques to address computer program complexity are of timely interest.

Modern programming methods seek among other things to alleviate software complexity through constraints on control transfer decision logic (Ref. 3). Furthermore, the resultant software is rendered more amenable to analysis because of the order imposed on the construction of the computer program (Ref. 4). The fundamental thrust of both of these software engineering initiatives has been to produce more reliable software whose quality is easier to confirm.

Terminology

Software used in computers which perform specific system functions (e.g., flight control) apart from those associated with general purpose computer facilities is termed embedded software. Desirably, it is the product of software engineering, which is the practical and methodical application of science and technology in the design, development, evaluation, and maintenance of computer programs. Establishing and confirming that this software properly fulfills its intended purposes under faulted and fault-free conditions is realized through the proper application of assurance technology. This is largely achieved through the V&V process; distinctions between validation and verification are noted in Table III. Note that not only is it possible and commonplace to validate a system, but a concept or a control algorithm as well. Similarly, it is normal to verify software specifications, test requirements, and modules as well as the complete computer program.

Problem

The basic problem addressed herein is that of validating flight-critical DFCS functions in the context of farther-term implementations. This focuses upon quality and safety associated with more complex flight software as might result from fault-tolerant, highly integrated, modern control oriented system implementations. The intent is to anticipate and address future assurance technology needs such that the V&V process for advanced DFCS can be sufficiently developed and purposefully accommodated in future system engineering methodologies.

Resolution of these V&V problems for flight-critical DFCS will become increasingly urgent because of concerns which previously have not applied to automatic flight control systems for production aircraft. Previously, an aircraft could be delivered and deployed with discrepant functions disabled, as for example an inhibited autoland capability. In the case of a pure digital fly-by-wire (DFBW) system, however, disabling its function totally disables the aircraft. Thus, the penalties for failing to develop and validate a DFCS are becoming much more severe.

Decomposition of the basic problem yields a seemingly non-consequential one of opting to adapt or develop selected software engineering techniques to flight control design practices, or to tailor the FCS development process to the use of repertoires of general purpose software tools. The former course is pursued here because full understanding of the application needs has repeatedly been found vital to the success of software implementation projects (Ref. 5). This should hold for specialized software techniques as well, at least with regard to analytical FCS engineers who routinely develop and use automated tools. The essential problem then is for flight control engineers to employ more powerful software engineering methods oriented toward DFCS needs in a systematic manner, and to leave the more general or esoteric issues to computer scientists.

Goals and Challenges

In summary form, the following elements of the general problem are seen as pivotal in achieving a software engineering based validation process which is capable of adequately fulfilling the assurance requirements levied on flight-critical DFCS:

- o User Acceptability - projected integrated flight stations with multifunction displays and automated crew assist functions for future operational environments pose user interface validation problems of substantial concern. Similar type problems in mission avionics have previously been found to be quite formidable (Ref. 6).
- o Conclusiveness - establishing compliance with quantitative safety and mission reliability requirements such as those of MIL-F-9490D remains an unresolved issue. Although such analyses lie outside the DFCS V&V process in the strict sense, system validation must support such analyses and confirm system safety under faulted and fault-free conditions.
- o Bounded Effort - increased flight criticality necessitates higher assurance levels, and hence a more intensive validation effort. For a practical DFCS development, however, there must be realistic bounds on the efforts and resources expended. Not unrelated to conclusiveness, bounding of effort must be based on measures of the scope and significance of the validation activities.
- o Minimal Risk/Schedule Span - reduced risk indicates the need to emphasize the front-end activities in the development cycle. Viability of system concepts and availability of the enabling technology are most important. Schedule is also favorably affected by front-end priorities and planning relative to technological resources.
- o Life Cycle Support - changes to DFCS seem inevitable, both during development and operational service. Responsiveness in effecting and confirming software modifications is important then, especially in achieving the cost/performance benefits which to some extent motivate digital mechanization.

PERSPECTIVE

As previously suggested, the perspective employed in this paper is that accruing from flight controls experience, yet tempered by acknowledgement of software engineering techniques. This view espouses a V&V methodology not radically different from current DFCS practice, but one which is bolstered to accommodate more demanding validation requirements. Consider then how traditional FCS practices can be augmented or enhanced by certain software techniques as a means of upgrading DFCS assurance technology.

Traditional FCS Practice

Analyses and testing are central to conventional FCS development efforts. Performance analyses usually focus on the control laws, and safety analyses on the organization of the system. Testing is performed in a sequence of activities which initially focuses on control laws and increasingly on the total system mechanization. Such practices continue to be effective for current digital mechanizations for several reasons:

- o System requirements are well understood
- o System configurations are relatively straightforward
- o Criticality of automated functions is quite limited.

Projected DFCS Practice

To meet the needs of future DFCS, it is projected that traditional FCS engineering practices will be augmented in the following ways. Note that increased system integration is presumed, as discussed in References 7 and 8, and that the front end of the system development cycle is accorded due emphasis.

- o Hardware considerations will be subordinate to software decisions - the "software first" approach described in Ref. 9 will be necessary to accommodate new or ill-defined user requirements, to validate system concepts, to develop crucial algorithms, and to define computer subsystem requirements. User interfaces and real-time interaction will be dominant concerns.
- o More assurance features will be designed into flight software - a variety of features which foster quality or assurance thereof will be designed into software. Most are based on modern programming practices which render software more testable and analyzable. Others relate to integral redundancy which foster consistency or reasonableness checks (Ref. 10).
- o Software analysis will be performed extensively - analysis of flight software will be widely employed because of its cost effectiveness and its value to an integrated methodology.
- o Higher order language (HOL) will continue to be increasingly important - the need for machine independence, i.e., software portability, in the "software first" approach and the capacity to design in assurance features will both further underscore the value of HOLs. This will not, however, preclude assembly language coding of target machine related functions.
- o Testing will be designed to elicit definitive and conclusive information - careful formulation of requirements and software analysis will be employed in determining optimal test strategies of pre-established significance. Several dimensions of test coverage will be employed to facilitate quantitative indices of test conclusiveness. Of the projected practices listed here, this is the most formidable in terms of requisite technology advances.

Software Engineering Initiatives

A consensus of distinguished leaders in software engineering concluded that there is "an increasingly urgent need ...(for)...testing and evaluating the many software engineering ideas and techniques that have been put forth" (Ref. 11). For reasons differing from those of the panel members, who wish to see the results of their research put to practical use, flight control engineers might well concur. The most compelling reason is simply that of the need for more powerful methods and tools to cope with increasing software complexity and flight criticality.

The cost and efforts involved in selecting and adapting software engineering assurance methods not only appears to be a matter of necessity, but one justified by the economics of not obtaining new methods. It should be noted that the methods for DFCS need not be elaborate or universally applicable. Rather they should respond to the specific needs of DFCS software, and to a surprisingly significant extent, these methods might be disguised or expanded versions of technology previously employed in flight controls. Development of a V&V approach will hopefully clarify this correspondence.

APPROACH

The approach to DFCS validation developed in this section is not a review or restatement of V&V methods or methodology. References 12, 13, and 14 already provide excellent surveys of the basic software assurance technology, especially with regard to verification techniques. Rather the intent here is to address far-term DFCS validation needs mainly from a flight controls background, yet from a much broader perspective. Thus, the approach formulated does not hesitate to incorporate software engineering type concepts so long as they promise to contribute to stated needs.

Recall that contemporary DFCS, e.g., digitized versions of prior analog systems, are not at issue; existing assurance technology is considered adequate to accommodate them. Of interest then are fault-tolerant DFCS performing augmented FBW and active control functions while interfacing with the flight crew via an integrated flight station. Thus, validation is addressed in terms of reconfigurable flight-critical systems whose architectures are not preconceived or restricted.

Hence the needs to focus on the front end of the system development process and to acknowledge the need for technical planning cannot be avoided. The emphasis, however, is on the essential technology and its proper utilization rather than on system management practices. Quite briefly then, the issues addressed here are how might the DFCS validation problems of the far-term be addressed and resolved, and what enabling technology remains to be developed?

System Development

In developing highly reliable systems of even moderate performance complexity, the confirmation of user acceptability must be considered from the outset along with performance and functional design issues. This is consistent with the old cliché that quality and reliability cannot be added after system design. Not only must quality be designed into the system, but ideally provisions for confirming quality and reliability should be devised and effectuated at appropriate steps of the development process. It is helpful, then, to consider the development process with emphasis on those activities which most significantly affect system validation.

A basic system development process compatible with Air Force Regulation 800-14 (Ref. 15) is depicted in Figure 1. This diagram emphasizes the software tasks and indicates several software verification steps. The process illustrated is considered quite acceptable in dealing with contemporary systems. In dealing with far-term DFCS, however, rather significant elaboration is essential. In this context, the Conceptual, Definition, Test & Integration, and Validation Phases are of particular interest.

Figure 2 is a simplified version of the system development process which highlights the activities of primary interest. For a truly advanced DFCS, the most crucial and challenging activity is usually that of defining system requirements which are realizable and which suitably support fulfillment of the mission requirements. In the Conceptual Phase then, a feasible conceptual system must be formulated to respond to evolving system requirements. At this point it is essential to ascertain that a practical system can be implemented which meets user needs.

During the Definition Phase, those requirements must be refined and finalized, but this can be deceptively difficult to do. First of all, the user needs may not be adequately determined, and secondly, they may be hard to express properly. Note that the substantial problems currently encountered in the form of changing requirements tend to become quite amplified for the type DFCS under consideration. Factors which can impede the determination of system requirements include: ill-defined flight crew roles, needs, and interfaces; inadequate airframe models or mission scenarios; and ill-defined or unsatisfactory interfaces with other systems. Where certain aspects of system requirements cannot be resolved, accommodations should be made for projected options, which must not presume that all problems can be resolved later through software changes.

Once properly defined, the system requirements must be stated in such a way that meaningful determination of compliance can ultimately be achieved and that they can be readily used to develop the requirements for the software and various hardware components. Unfortunately, difficulties with system requirements, which can arise from erroneous, inadequate, or ambiguous provisions, tend to propagate to the component specifications. These types of deficiencies are all too common, and their correction relatively costly.

Shifting to the output end of the development process, consider next the nature and problems of system validation. Note that software verification constitutes an inner loop function to be completed, along with system integration and development testing, prior to the initiation of the validation activity. System validation relates directly back to the system requirements, and if the Definition Phase has been properly concluded, validation normally proceeds much more smoothly.

System validation does involve getting the user back in the loop, normally in a much more tangible manner. Thus, serious problems in user acceptability may arise, if only because more data are available to the evaluators. But the prerogative to demonstrate compliance with the finalized system requirements should exist, and addressing this problem is a technical matter of immediate concern.

Basically, the intent of system validation is to confirm that a DFCS satisfactorily performs all of its intended functions without any undesired side effects. The term system is construed to encompass hardware, software, flight crew, and procedures as they interact over the range of admissible flight conditions. Demonstrating proper operation of explicitly described functions is clearly a more straightforward undertaking than confirming that under no untoward circumstances might the system exhibit undesired behavior. This is particularly true for digital mechanizations, where the structure and organization of the software is relatively complex compared to analog equivalents. In any event, system validation must address these two aspects of system operation, normal and anomalous, and do so in an efficient and conclusive manner.

Integrated Methodology

An integrated V&V methodology is one in which the various assurance methods contribute in a timely and complementary manner to provide adequate overall coverage. Defining and evaluating coverage, however, is a significant problem in itself, as is a meaningful interpretation of its significance relative to system acceptability. Once an approach to coverage is formulated, the methodology should provide an optimized strategy for enhancing coverage for the time and effort expended. This in turn is based upon software analyses and automated testing.

Such a methodology is compatible with the development process depicted in Figures 1 and 2. The course then is to develop in more detail certain aspects of the process. Unfortunately, existing technology has not been found adequate to effectuate the overall methodology, so the associated research needs are identified in this section and discussed in a subsequent one.

As noted in Figure 2, the concept and the system validation activities are of primary interest here. Concept or control law algorithm validation is undertaken to minimize risk by clarifying and defining system requirements and implementation decisions at the front end of a project. Both involve simulation, possibly of a very substantial extent. The associated technology however, is not unavailable. The crucial issue is the recognition of the need or value of this stage of validation, which is not often pursued adequately, and the resultant allocation of resources.

Ironically, the system validation activity, which is rather customary, is considered to be in appreciable need of technology advances to accommodate future DFCS. Again, simulation is seen to be a vital part of the process, owing in part to the more important role of the flight crew in validating innovative configurations. Table IV summarizes the attributes of these three types of validation.

As a prelude to further consideration of system validation, assume the following scenario:

- o System requirements are documented
- o The embedded software is designed well and is verified
- o Design assumptions are clearly identified
- o DFCS integration and development is completed.

The overall objective of system validation can then be expressed in somewhat more detail by the following expression:

$$\text{VALIDATION} = \underbrace{(\text{PROGRAM COVERAGE}) (\text{METHODS RELIABILITY}) + (\text{ERROR COVERAGE}) (\text{PROGRAM COVERAGE}) (\text{METHODS DEPENDABILITY})}_{\text{Confirm No Anomalies}}$$

Affirm Acceptability

This expression indicates the twofold nature of validation. In the first case, rather strong conclusions regarding system acceptability are sought. This is very challenging, even where testing is supported by suitable analyses. Increasing the likelihood of detecting software errors constitutes the thrust of the second aspect. This involves prior assessment of the types of potential software errors and of effective testing means of determining actual occurrences. Note that both aspects of system validation are basically test oriented, as appropriate to questions relating to externalized phenomena, such as user environment and procedures. Strictly analytical means are considered more germane to software verification.

In this context, the approach is to devise test strategies which purposefully elicit software error manifestations when applied. Note that although software receives special attention, the basic emphasis is on the acceptability of the total system. Here software performs system functions through interactions with hardware and the flight crew, and these functions must fulfill the stated system requirements. These in turn, then, must be translated into explicit test procedures which properly acknowledge details of the DFCS implementation. Figures 3 and 4 summarize this process, which presumes the availability of the various validation methods to determine the test sequence.

This type of process is not incompatible with recent software engineering research. One such initiative of particular relevance has sought better ways of relating the input domain and software structure to test case generation strategies. While basic theoretical work has been done in this area (Ref. 16), meaningful extension to practical methods remains to be achieved. The promising benefits of improved test coverage, conclusiveness, and efficiency continue to motivate interest. The cases most easily dealt with are those of discrete-valued input domains. In fact, when the number of possible combinations of input values is relatively small, exhaustive testing of all possibilities is a potentially realizable approach to verification or validation.

Test case generation strategies are not well-developed, however, for input domains encompassing a large number of discrete-valued variables or even a small number of continuous variables. Better methods could substantially reduce the huge number of simulations often required to gain confidence in the system over the flight envelope.

Figure 5 represents a recommended approach toward developing those software engineering techniques which are considered vital in attaining assurance levels to cope with flight-critical functions. These techniques center largely on software analysis techniques, and those associated with the negative (error detection) aspects of validation presently seem reliable for given implementations. Those associated with the affirmative (proof equivalency) aspects merit considerably more applied research attention. Quantitative measures of program coverage and the determination of test significance are of pivotal importance. These vital issues are examined further in a subsequent section.

Optimized Testing

Testing can be optimized through improved test procedures, facilities, and termination criteria. Properly stated requirements and software analyses constitute the basis for improved procedures as indicated in Figure 3, but further consideration needs to be given the nature of the software analyses. Facilities to enhance testing include automatic stimuli insertion, execution monitoring, and results processing. These are all based on suitable internal access to the flight computers during program execution; the major concern is to interpret the stimuli correctly and not to affect normal operation in any unintended way. Table V indicates the similarity of this software testing technology with prior FCS technology. In the case of termination criteria, software analysis again provides the basis for the decision.

Software analysis plays a vital role in DFCS testing, because black box type functional testing is inadequate to exercise the flight software properly. Through careful examination of the program, analyses can provide meaningful test strategies. A thrust of affirmative testing is confirming the absence of specific potential deficiencies, rather than just exercising the system over some portion of its input domain. Collectively, these test cases confirm the absence of errors.

Affirmative testing must consider the input domain, the design of the implementation, and the functional requirements related to the domain. Consideration of the input domain is essential in that all of the input possibility are germane to the formulation of meaningful test cases. This must include the input variables and the range of possible values of each, as well as the combinations of values which are possible in the operational environment. For DFCS, the input domain includes Boolean variables, such as the mode or validity signals, and sensor inputs which are discrete valued as a result of analog-to-digital (A/D) conversion.

The functional requirements defined on the input domain are essential in that these tend to provide a starting point for partitioning the input domain as well as to form the basis for evaluating the test results. The Boolean variables, at the system input level, are generally used in the selection of internal flow paths, and so the partitioning effect is usually evident. Affirmative tests set up on such variables are formulated to confirm that the desired paths are in fact being selected internally.

Certain sensor variables, such as altitude or Mach, are also used in flow path selection. Examples of this are in functions, e.g., gain scheduling, which are different over certain ranges of altitude or Mach. The threshold values of such sensor variables also provide insight into meaningful partitioning of the input domain. The affirmative testing must confirm that the path selection logic correctly uses the values from various regions of the partitioned input domain.

The internal structure and design of the implementation must also be used in partitioning the input domain. Singular points resulting from the computational algorithms are of interest, as well as flow path selection which occur in the case of a non-linear gain schedule function represented by several piecewise approximations. The external definition of the schedule might be by a table giving selected values. The table itself would give no indication of the internal logic involved in determining the gain; in this example, the selection of the proper piecewise approximation.

As mentioned previously, affirmative testing must also relate to the functions performed on the input domain, so test cases are required which confirm the generation of the acceptable outputs. The confirmation that the outputs are acceptable for the input test cases considered is fairly straightforward; establishing the freedom from any undesired effects is somewhat more challenging.

This type of testing is rather common programming practice wherein special tests are devised to specifically verify some portion of a program, such as a complex segment of control logic. Frequently this is done on an intuitive or informal basis, without rigorous analysis or documentation of the types of errors that the test would reveal. Nonetheless, the test is oriented toward affirming correctness, and it is devised with the program design very much in mind.

Tests of large computer programs tend to be oriented somewhat differently. The nature of the input domain and the functional requirements are considered in generating test cases, but the software design may have reduced influence on the test cases selected.

A challenge for FCS engineering, then, is to utilize the software engineering contributions to test case selection, theory, and to extend and adapt them to systems applications. Table VI lists an integrated collection of methods which might be employed during DFCS validation, and indicates their intended purposes.

CONCLUSIONS

- o User acceptability of farther-term DFCS will increase the importance of simulation, both on the front end and the tail end of the development cycle.
- o Determination of detailed DFCS requirements will be much more important than a formalized expression of them.
- o Full-time flight criticality and increased complexity will compel the adoption of software engineering based assurance methods.
- o Software analyses will play a vital role in defining DFCS test strategies and procedures.
- o Software engineering issues of program coverage, test conclusiveness, and termination criteria should be further investigated in the context of DFCS needs.
- o Flight software error data bases should document the specific V&V measures which the respective errors have evaded.

REFERENCES

1. MIL-F-9490D, "Flight Control Systems - Design, Installation and Test of Piloted Aircraft, General Specification For," U.S. Air Force, 6 June 1975.
2. FAA Advisory Circular AC-20-57A, "Automatic Landing Systems (ALS)," U.S. Department of Transportation, Federal Aviation Administration, Washington, D.C., 12 January 1971.
3. Mills, H. D., "Mathematical Foundations for Structured Programming," IBM FSC 72-6012, Gaithersburg, Maryland, February 1972.
4. Feign, D., "Computer Science, Software Engineering, and the Cost of Software," Computer, December 1978.
6. Morgan, L. F., "S-3A Avionics: Software Revolution Forerunner," Journal of Aircraft, January 1975.
7. Ostgaard, M. A., "Control Sciences and Technology Capability Potential," AFFDL/FG Memo, Wright-Patterson AFB, Ohio, June 1978.
8. Ellison, T. A., "Airline Viewpoint on Systems Development and Integration," AIAA 2nd Digital Avionics Systems Conference, Los Angeles, California, November 1977.
9. Boehm, B. W., "Software and It's Impact: A Quantitative Assessment," Datamation, May 1973.
10. Saib, S. H. et al, "Software Quality Laboratory," Computers in Aerospace Conference, Los Angeles, California, October/November 1977.
11. Wasserman, A. I. et al, "Software Engineering: The Turning Point," Computer, September 1978.
12. Fujii, M. S., "Independent Verification of Highly Reliable Programs," COMPSAC 77, Chicago, Illinois, November 1977.
13. DeWolf, J. B. and J. Wexler, "Approaches to Software Verification with Emphasis on Real-Time Applications," Computers in Aerospace Conference, Los Angeles, California, October/November 1977.
14. Osterweil, L. J., "A Methodology for Testing Computer Programs," Computers in Aerospace Conference, Los Angeles, California, October/November 1977.
15. AF Regulation 800-14, Vol. II, "Acquisition and Support Procedures for Computer Resources in Systems," Headquarters USAF, Washington, D.C., 16 September 1975.
16. Goodenough, J. B. and S. L. Gerhart, "Toward a Theory of Test Data Selection," 1975 International Conference on Software Reliability, Los Angeles, California, April 1975.

TABLE I. MOTIVATING FACTORS

Trend	Need	Solution	Remarks
Increased Demands on Crew	Reduced Crew Workload	Automate/Optimize Cockpit Activities	Two-Man Crew Complicates Problems
Denser Airspace Utilization	Improved Air Traffic Control	Enhanced Pilot/Controller Interaction	Cockpit Displays/Procedures Affected
		More Precise Position Fixing	Global Position System Reference
		Airborne Collision Avoidance	System Must be Operationally Practical in Cockpit
Unacceptability of Weather Delays	More All-Weather Capability	All-Weather Take-off/Landing	Microwave Landing System Options
		Improved Weather Information	Weather Updating and Display
		Severe Windshear Avoidance	Windshear Detection and Warning
More Economical Operations	Fuel Conservation	Active Controls	Multiplicity of Functions
		Optimal Energy Management	Guidance Path/Throttle Settings
	Better Dispatchability	Automated Maintenance	Sophisticated Self-Test Capability
		Better Resource Utilization	System Fault-Tolerance and Reconfiguration
Environmental Concerns	Noise Abatement	Selected Climbout/Descent Profiles	Guidance/RNAV Considerations
	Emissions Control	Engine Monitoring/Tuning	Maintenance/Performance Implications

TABLE II. IMPLICATIONS OF ADVANCED CONFIGURATIONS

PROSPECT	CONSEQUENCE	RESULTANT NEEDS	
		SYSTEM	METHODS
Extensive Deployment of Flight-Critical Functions	Higher Levels of Safety Assurance Required	More Reliable System Mechanizations	More Powerful Assurance Methods
Tailored System Architectures	Complex Executive Programs	Improved System Engineering Practices	More Versatile Assurance Methods
More Functional Integration	Complex Applications Programs	Improved Software Engineering Practices	More Powerful Assurance Methods
More Intense Operational Environment	Increased Pilot Workload	More Pilot Assist Functions	Human Factors Workload Methods
Integrated Flight Station	Information Management Distillation	Computer Processing and Formatting for Displays	Human Factors Communication Methods

TABLE III. DISTINCTIONS BETWEEN VALIDATION AND VERIFICATION

CHARACTERISTICS	VALIDATION	VERIFICATION
Scope	Total System	Computer Program
Reference	System Requirements	Software Requirements
Emphasis	Overall Acceptability	Logical Correctness
Focus on Software	Externalized Behavior	Internalized Details
Environment	User or Simulated	Test or Abstract

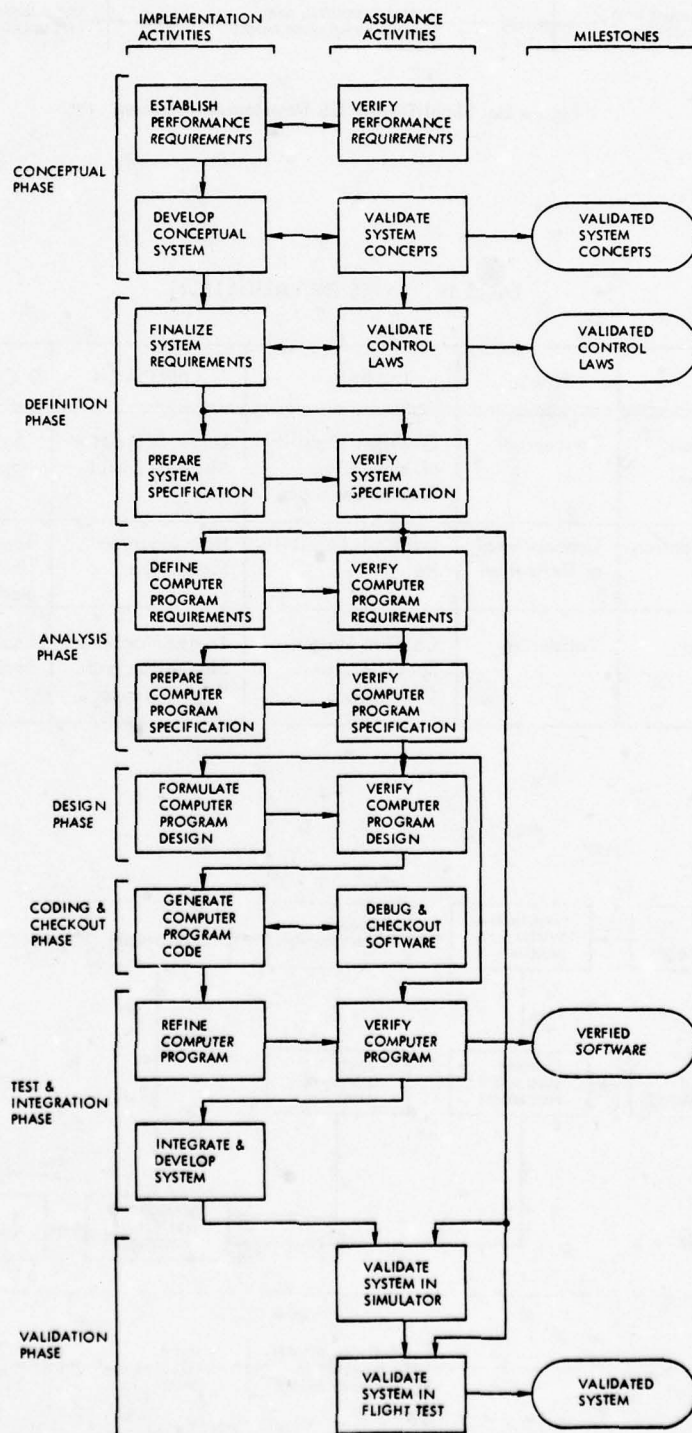


Figure 1. DFCS Development Process

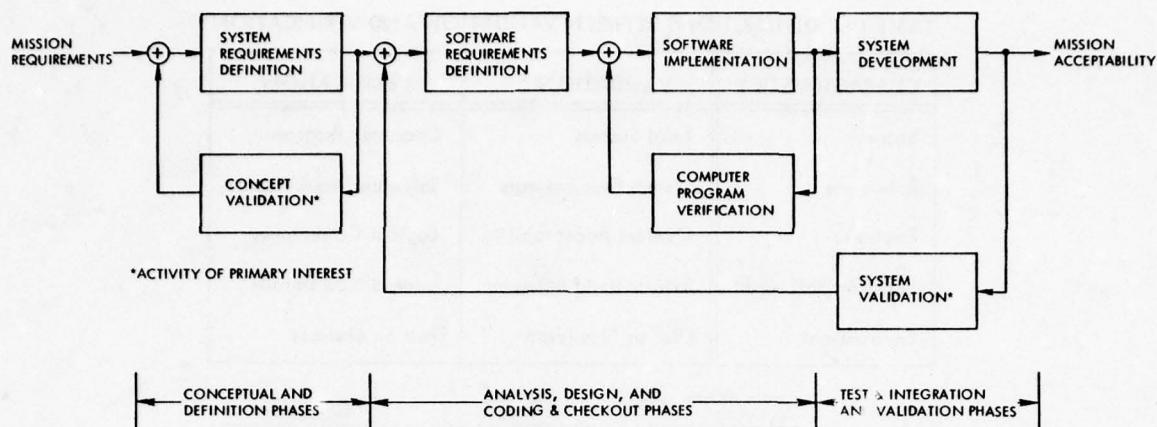


Figure 2. Simplified DFCS Development Process

TABLE IV. TYPES OF VALIDATION

TYPE	PHASE	INTENT	APPROACH	COMMENT
Concept Validation	Conceptual	Establish Viability of Innovative System Concepts	Large-Scale Host Machine Simulation	"Software First" Approach
Control Law Validation	Conceptional or Definition	Establish Feasibility	Host Machine Simulation	Top-Down/Bottom-Up Software Development
System Validation	Validation	Confirm System Specification Compliance	Target Machine Simulation and Flight Testing	Usually Intended Sense of Validation

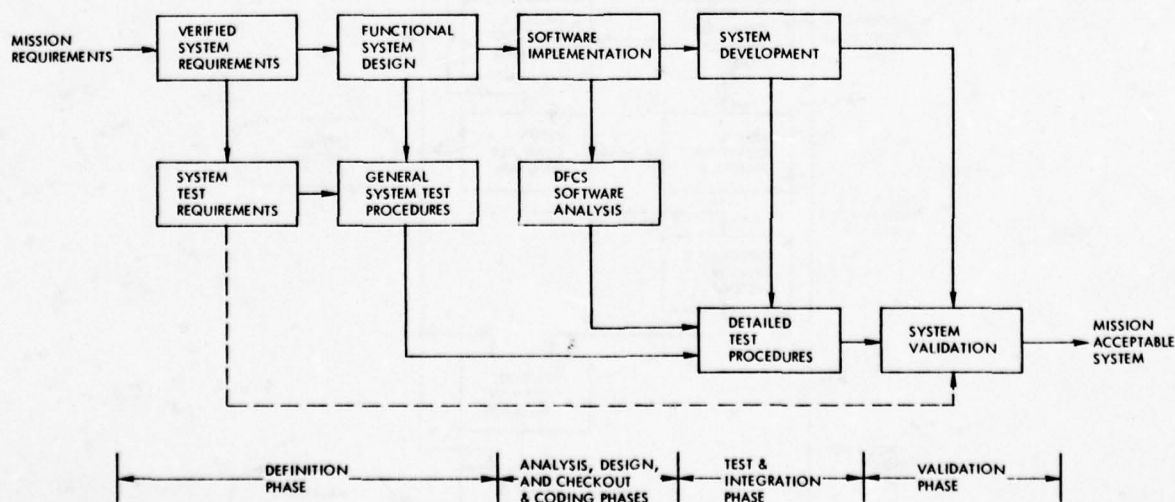


Figure 3. Test Procedure Generation Overview

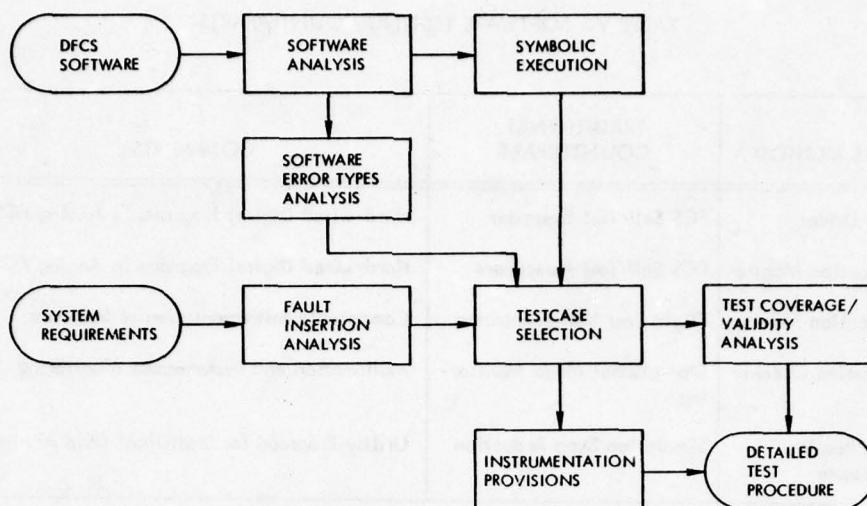


Figure 4. Detailed Test Procedure Generation

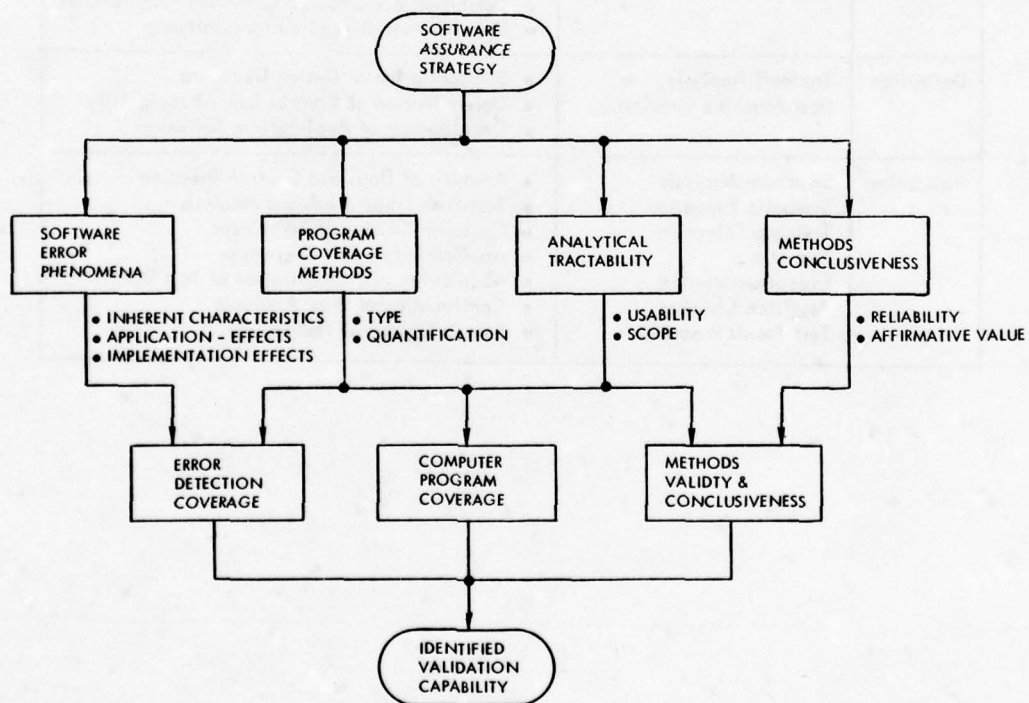


Figure 5. Validation Technology Approach

TABLE V. SOFTWARE TESTING COUNTERPARTS

TEST METHOD	TRADITIONAL COUNTERPART	COMMENTS
Test Driver	FCS Self Test Exercises	Hard-wired Digital Programs in Analog FCS
Execution Monitor	FCS Self Test Assessment	Hard-wired Digital Programs in Analog FCS
Execution Monitor	Flight Test Instrumentation	Comparable Instrumentation of Software
Assertion Checker	Operational Mode Monitoring	Malfunction and Performance Monitoring
Test Result Processor	Simulation Data Reduction	Utility Programs for Statistical Data Analysis

TABLE VI. DFCS VALIDATION METHODS

PHASE	METHOD	CONTRIBUTION
Conceptual	Large-Scale Host Machine Simulator	<ul style="list-style-type: none"> • Determination of Concepts Feasibility • Definition of Flight Crew Interface • Refinement of System Requirements • Definition of Computer Subsystem Requirements • Development of Applications Software
Definition	Tradeoff Analysis Host Machine Simulation	<ul style="list-style-type: none"> • Substantiation of Design Decisions • Determination of Control Law Acceptability • Development of Applications Software
Validation	Structure Analysis Symbolic Execution Testcase Selection Test Driver Execution Monitor Assertion Checker Test Result Processor	<ul style="list-style-type: none"> • Analysis of Data and Control Structure • Test Path Generation and Analysis • Optimum Choice of Test Cases • Application of Test Sequences • Acquisition and Compilation of Test Data • Confirmation of Prior Analyses • Summarization of Test Results

LOGICIEL AVIONIQUE:

EXPERIENCES PRATIQUES D'UNE METHODOLOGIE

J. PERIN

ELECTRONIQUE MARCEL DASSAULT

92214 SAINT-CLOUD

FRANCE

RESUME : Cet exposé décrit l'organisation et la méthodologie appliquées pour la réalisation des logiciels implantés dans les calculateurs principaux des avions MIRAGE F1 et MIRAGE 2000.

L'accent est plus particulièrement mis sur l'importance fondamentale des phases de définition et de validation des programmes.

Des exigences de sûreté de fonctionnement élevées, alliées à de sévères contraintes industrielles ne permettent pas de prendre des risques considérables en expérimentant "à chaud" des méthodes non confirmées.

C'est pourquoi la caractéristique principale du développement de ces logiciels a été la mise en oeuvre de techniques et d'outils traditionnels dans le cadre d'une méthodologie rigoureuse et par un personnel à fort potentiel.

CHAPITRE I

INTRODUCTION

L'ELECTRONIQUE MARCEL DASSAULT (EMD) est spécialisée dans l'étude, le développement et la fabrication d'équipements électroniques de pointe, tant dans le domaine militaire que dans le domaine civil.

L'effectif de l'EMD est de 2500 personnes, dont 1400 ingénieurs et cadres. L'informatique aérospatiale (calculateurs, bus numériques, systèmes digitaux, logiciels de base et d'application) constitue une des activités principales de l'ELECTRONIQUE MARCEL DASSAULT : 20 à 25 % du chiffre d'affaires est réalisé dans ce domaine.

Depuis 1965, époque à laquelle l'EMD a conçu le premier calculateur embarqué européen utilisant des circuits intégrés, les missiles balistiques français sont équipés de calculateurs universels EMD, puis EMD-SAGEM à la suite d'accords de coopération signés entre les deux sociétés.

En 1976, l'accroissement des besoins en matière de puissance de calcul conduit l'EMD à promouvoir en France de nouvelles technologies de composants et de circuits pour créer une nouvelle génération de calculateurs universels :

- 1084 pour les missiles balistiques,
- M182 pour l'avion MIRAGE F1,
- 2084 pour l'avion MIRAGE 2000.

Le système de transmission des informations numériques à bord de ces missiles et avions a lui aussi été développé par EMD : c'est le bus numérique GINA.

L'EMD réalise également tous les logiciels de base et, sous la maîtrise d'oeuvre de ses clients, la plupart des logiciels d'application concernant ses propres calculateurs aérospatiaux. L'introduction de la nouvelle génération de calculateurs EMD en tant que calculateurs principaux des avions MIRAGE F1 et MIRAGE 2000 développe considérablement cette activité logiciel : le volume des programmes à définir et à réaliser dans les trois domaines des logiciels de base, des logiciels d'application, des logiciels de validation a nécessité la constitution progressive d'un groupe d'environ 60 ingénieurs.

Une méthode de travail extrêmement rigoureuse a été mise en place pour gérer l'ensemble de ces projets. Avant de décrire les grands principes de cette organisation, les enseignements pratiques résultant de son application depuis plus de deux années et les améliorations à y apporter, il est indispensable de préciser le contexte général de cette expérience méthodologique par :

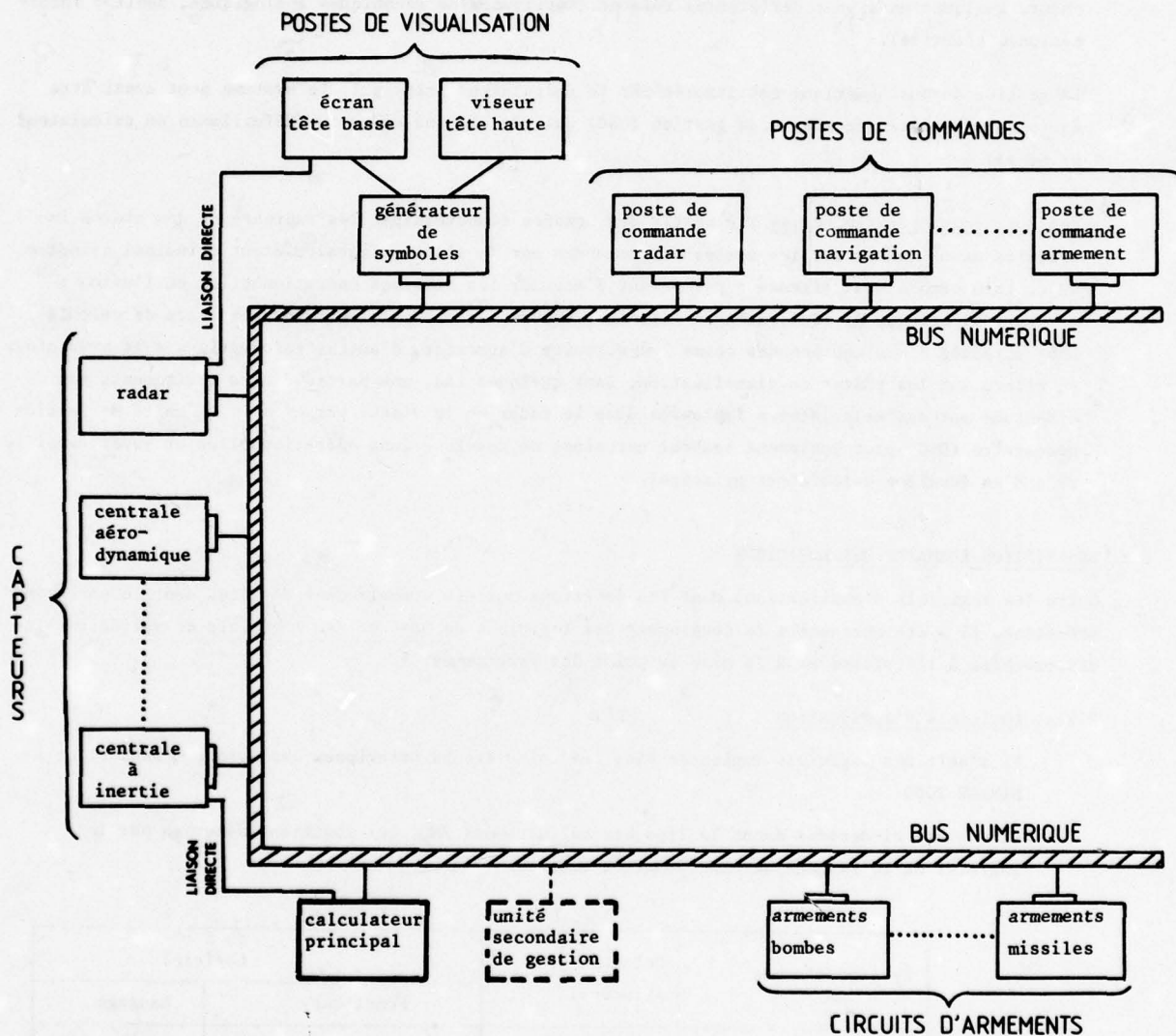
- a) un bref rappel des fonctions du calculateur principal dans un système d'armes avionique,
- b) la description sommaire des différents logiciels,
- c) un aperçu des contraintes industrielles de ces réalisations.

CHAPITRE II

----- CONTEXTE DE L'EXPERIENCE METHODOLOGIQUE -----

2.1. FONCTIONS DU CALCULATEUR PRINCIPAL DANS UN SYSTEME D'ARMES AVIONIQUE

Le système de navigation et d'attaque (SNA) d'un avion de combat moderne peut être très schématiquement illustré, dans son principe général, par la figure suivante :



Parmi les principaux équipements du SNA, on distingue :

- des capteurs : centrale à inertie, centrale aéro-dynamique, radar, etc...
- des postes de visualisation : écran tête-basse, viseur tête-haute, etc...
- des postes de commande : pour la navigation, l'armement, le radar, etc...
- des circuits d'armements : pour bombes, canons, missiles, etc...

Bien que tous ces équipements comportent une part de plus en plus grande d'électronique numérique, c'est à dire de processeurs spécialisés et très intégrés au matériel, les calculs effectués y sont de nature différente de ceux du calculateur principal : ils sont spécifiques de l'équipement, contribuent directement à ses performances et ne traitent généralement que des données locales.

Au contraire, le logiciel du calculateur principal intervient au niveau global du système de façon relativement indépendante des caractéristiques particulières des équipements. Il assure deux types de fonctions :

- des fonctions de gestion centralisée (échanges d'information, surveillance du fonctionnement d'ensemble) :

Le schéma fait apparaître le rôle particulier du bus numérique ou "digibus" auquel sont connectés la plupart des équipements du système d'armes. Les informations échangées entre ces équipements transitent sur cette liaison sous forme numérique et suivant un mode de multiplexage temporel à haute fréquence. Les liaisons directes entre équipements sont de plus en plus rares : il en subsiste encore quelques unes pour différentes raisons (survivance de techniques analogiques, débit d'informations, sécurité).

La gestion du bus numérique est assurée par le calculateur principal. Le système peut aussi être équipé d'une unité secondaire de gestion (USG) qui gère le bus en cas de défaillance du calculateur principal.

- des fonctions opérationnelles : à partir des données élaborées par les capteurs et des ordres introduits manuellement sur les postes de commandes par le pilote, le calculateur principal effectue un certain nombre de traitements permettant d'assurer les missions opérationnelles de l'avion : navigation, attaque Air-Sol, attaque Air-Air ; suivant la mission, certains résultats de calculs sont adressés à des équipements comme les circuits d'armement, d'autres informations sont présentées au pilote sur les postes de visualisation. Dans quelques cas, une partie de ces traitements est effectuée par des calculateurs implantés dans le radar ou le viseur par exemple. L'unité de gestion secondaire (USG) peut également assurer certaines de ces fonctions opérationnelles et jouer ainsi le rôle d'un deuxième calculateur principal.

2.2. DESCRIPTION SOMMAIRE DES LOGICIELS

Outre les logiciels d'application, dont les fonctions ont été sommairement décrites dans le paragraphe précédent, il a été nécessaire de développer des logiciels de base et des logiciels de validation indispensables à l'écriture et à la mise au point des programmes.

2.2.1. Logiciels d'application

Il s'agit des logiciels implantés dans les calculateurs principaux des avions MIRAGE F1 et MIRAGE 2000.

Le tableau ci-dessous donne le type des calculateurs EMD, les fonctions assurées par le logiciel et le langage de programmation retenu.

Système d'armes	Calculateur principal	Logiciel	
		Fonctions	Langage
1er type MIRAGE F1	M182	a) gestion centralisée b) Navigation + Air-Sol	Assembleur
2ème type MIRAGE F1	M182	a) gestion centralisée b) Navigation + Air-Sol	Assembleur
1er type MIRAGE 2000	2084 + USG 284	a) gestion centralisée b) Navigation + Air-Sol + Air-Air	LTR
2ème type MIRAGE 2000		a) gestion centralisée b) Navigation + Air-Sol + Air-Air	LTR

2.2.2. Logiciels de base

Ont été réalisés les logiciels de base suivants :

- a) sur ordinateur universel (IBM) : assembleur, compilateur LTR (le LTR est un langage évolué temps réel défini en France par le Centre de Programmation de la Marine dès 1969 et qui a pour vocation d'être le langage de programmation temps réel commun aux trois armes : Terre, Mer, Air), simulateur, chaîne d'archivage et de gestion des bibliothèques de programmes.
- b) en local, sur les calculateurs embarqués eux-mêmes : assembleur local, moniteur temps réel multi-tâches, programmes d'aide à la mise au point du logiciel d'application, programmes de tests et de recette du matériel...

2.2.3. Logiciels de validation

Ces logiciels sont implantés dans le calculateur d'un matériel appelé "Baie de Validation du Logiciel" (BVL). Cette baie est utilisée pour tester dynamiquement et valider les logiciels d'application avant l'intégration effective du calculateur principal dans son environnement opérationnel.

Cet outil particulièrement performant simule les équipements réels constituant le système d'armes, non pas au niveau de leur fonctionnement intrinsèque, mais à celui de leur interface avec le calculateur embarqué, en respectant scrupuleusement les aspects temporels, informationnels et interactifs. La conception et les possibilités des "Baies de Validation du Logiciel" développées par EMD sont précisées dans le chapitre suivant (paragraphe 3.6.6).

Le logiciel qui a été réalisé comporte un tronc commun (20 K mots) et une partie plus proprement spécifique des équipements à simuler qui peut atteindre 35 K mots pour les systèmes de navigation et d'attaque où le rôle du calculateur principal est le plus important.

2.3. CONTRAINTES INDUSTRIELLES

L'étude et le développement de tous ces logiciels se sont déroulés dans un contexte caractérisé par un certain nombre de contraintes dont certains aspects ont déjà été évoqués :

- a) les délais étaient très courts : environ un an entre le début de l'élaboration des spécifications et les premières livraisons de logiciel au banc d'intégration du système d'armes. De ce fait, l'étude, la fabrication et la mise au point des calculateurs devaient s'effectuer simultanément à la définition et au développement du logiciel.
- b) les interfaces étaient difficiles à définir parce que beaucoup d'équipements devaient être modifiés ou même développés pour les besoins spécifiques de ces systèmes avioniques.
- c) l'outil de validation du logiciel (BVL) était entièrement à concevoir et à réaliser, tant au point de vue matériel que logiciel.
- d) un certain nombre de logiciels de base étaient à développer complètement : moniteur, compilateur LTR, chaîne d'archivage.

Si ces contraintes rendaient les projets difficiles, par contre il existait un certain nombre d'éléments favorables :

- a) nous disposions dès 1977 d'un ensemble de services logiciels regroupant au total plus de 150 jeunes ingénieurs possédant une bonne expérience dans les domaines du logiciel de base, du logiciel d'application temps réel et ayant de plus une formation générale en électronique, automatisme ou aéronautique. Cette réserve de potentiel humain a facilité la mise en place rapide d'équipes importantes et compétentes.

- b) d'assez nombreux points communs existaient entre les différents systèmes avioniques, ce qui, en spécialisant quelques ingénieurs sur des problèmes précis, a permis un certain transfert de personnel entre les projets en fonction de retards ou de difficultés techniques imprévus.
- c) le fait d'avoir confié la réalisation du bus numérique, des calculateurs, des logiciels et de leurs outils de production et de test à la même société a permis d'atténuer les difficultés inhérentes à toute mise au point simultanée de matériels et de logiciels.

CHAPITRE III

LA METHODOLOGIE

3.1. OBJECTIFS

De manière générale, la méthodologie apparaît comme une forme d'organisation technique, humaine et administrative du travail permettant de :

- définir, répartir, coordonner les activités d'un personnel nombreux,
- prévoir, mesurer, contrôler les délais, les coûts, l'avancement et la qualité des travaux.

Notre méthodologie a été plus particulièrement orientée vers la production d'un logiciel de grande qualité répondant aux trois objectifs suivants : maintenabilité, fiabilité, optimisation.

En effet, les caractéristiques du contexte qui viennent d'être évoquées laissaient entrevoir que le logiciel serait livré de façon partielle et progressive, que les adaptations apportées aux spécifications initiales seraient nombreuses, s'étendraient sur toute la durée des projets et devraient être prises en compte rapidement. C'est pourquoi il fallait que le logiciel ait une excellente maintenabilité, et donc qu'il soit structuré de façon à être facilement modifiable dans des délais relativement courts sans remise en cause de l'ensemble des programmes.

Le logiciel devait évidemment posséder une très bonne fiabilité pour assurer une bonne sécurité de fonctionnement du système avionique.

Enfin, comme dans tout logiciel temps réel embarqué, l'optimisation du logiciel d'application en charge de calcul et encombrement mémoire devait être assez poussée.

Ces trois objectifs (maintenabilité, fiabilité, optimisation) ne sont d'ailleurs pas compatibles : en particulier, la maintenabilité ne peut être obtenue qu'au détriment de l'optimisation des programmes. Un certain compromis est à établir, variable suivant les traitements à effectuer : nous avons pris comme règle que le coût en mémoire de la structuration modulaire des programmes ne devait pas dépasser 10 % et que le coût en charge de calcul devait rester inférieur à 3 %.

3.2. PRINCIPES GENERAUX

Des exigences de sûreté de fonctionnement élevées, alliées à de sévères contraintes industrielles, ne permettent pas de prendre des risques considérables en expérimentant à chaud des méthodes non confirmées. C'est pourquoi la caractéristique principale du développement de nos logiciels a été la mise en oeuvre de techniques et d'outils traditionnels dans le cadre d'une méthodologie rigoureuse et par un personnel à fort potentiel.

Les principes généraux appliqués ont été les suivants :

- a) décomposition des travaux en étapes caractérisées par des responsabilités clairement définies et des points de contrôle permettant d'en mesurer l'avancement.
- b) écriture d'une documentation préalablement à toute réalisation importante, une mise à jour étant faite en fin d'étape en fonction du travail réellement effectué.
- c) pour chaque équipe, interdiction de commencer une étape tant que la précédente n'est pas terminée.
- d) procédures de test et de validation très poussées.
- e) archivage et gestion des bibliothèques de programmes sur ordinateur.
- f) contact permanent avec le maître d'oeuvre du système d'armes afin de lever les ambiguïtés ou imprécisions des spécifications au fur et à mesure qu'elles sont détectées.
- g) à tout moment, possibilité de modifications à condition que cette prise en compte se fasse au plus haut des niveaux concernés (par exemple spécifications système, analyse informatique, codage des programmes, etc...) quel que soit le degré d'avancement du logiciel. Cette procédure est indispensable pour garder l'homogénéité de la documentation.

La méthodologie a été appliquée aux trois types de logiciels que nous avons précédemment distingués : logiciels d'application implantés dans le calculateur principal, logiciels de base, logiciels de validation. Pour rester dans le cadre des sujets abordés au cours de ce symposium, la suite de cet exposé traitera uniquement du développement du logiciel d'application.

3.3. PHASES ET ETAPES

L'ensemble des tâches logiciel a été décomposé en quatre phases, correspondant à des activités et des responsabilités assez différentes ; chaque phase comporte elle-même plusieurs étapes. Le coût du logiciel d'application a été ventilé en fonction de chacune de ces étapes (le prix du matériel de tests et de validation n'a pas été pris en compte).

1ère phase : DEFINITION

étape 1.1. : définition globale

étape 1.2. : définition détaillée

14 %

2ème phase : PLANIFICATION

étape 2.1. : réexamen de la faisabilité technique

étape 2.2. : réexamen des besoins

étape 2.3. : réexamen des coûts et des plannings

1 %

3ème phase : REALISATION

<u>étape 3.1.</u> : analyse globale	10 %
<u>étape 3.2.</u> : analyse détaillée	20 %
<u>étape 3.3.</u> : codage et tests unitaires	20 %
<u>étape 3.4.</u> : intégration et tests statiques	5 %
<u>étape 3.5.</u> : tests dynamiques et validation	30 %
	<hr/> 100 %

4ème phase : EXPLOITATION ET MAINTENANCE

- essais opérationnels au sol
- essais opérationnels en vol

Le fabricant du logiciel n'intervenant pas dans la définition globale du système, cette étape n'est pas, pour lui, génératrice de coûts. Par ailleurs, les travaux de la quatrième phase n'ont pas été évalués isolément mais répartis sur l'ensemble des autres phases ; ils résultent en effet d'adaptations apportées à la définition du logiciel et sont la source de coûts de même nature que ceux découlant directement des spécifications initiales.

3.4. PHASE DE DEFINITION

Cette première phase est primordiale, les insuffisances à ce niveau peuvent être la cause essentielle de dépassements en coûts et délais, quelle que soit la qualité des travaux ultérieurs. En particulier, les moyens de validation du logiciel ne sauraient en aucun cas pallier les lacunes et ambiguïtés des spécifications. Dans tous les projets d'une certaine complexité, la souplesse du logiciel, c'est à dire sa facilité apparente de modification comparée à celle du matériel, est plus souvent un danger qu'un avantage, car elle conduit trop fréquemment à considérer "qu'il est toujours trop tôt pour définir ce que l'on veut et jamais trop tard pour modifier ce qui existe", comme l'a écrit Monsieur VANDECASTEELE, responsable des problèmes d'intégration de systèmes d'armes aux services techniques aéronautiques de l'armée de l'air française.

Cette phase de définition si fondamentale a été placée sous la responsabilité du maître d'oeuvre du système avionique et réalisée en étroite collaboration avec les équipementiers ; y ont également participé pour avis ou approbation les pilotes d'essais et les clients.

Dans la première étape, les fonctions opérationnelles globales du système d'armes ont été définies dans une optique "utilisateur" ; elles donnent lieu à un document de "spécifications globales du système".

La deuxième étape, où la participation du fabricant du logiciel a été extrêmement importante, a eu pour objet d'analyser et de décrire sans ambiguïté les traitements à effectuer par le calculateur principal. Cette description a été conduite dans une optique de décomposition fonctionnelle plus qu'opérationnelle. Ces travaux se sont concrétisés par la rédaction de "spécifications détaillées du logiciel", document contractuel du fabricant du logiciel vis à vis de son client.

Après coup, il est apparu que cette phase de définition devait être structurée de façon différente : introduction d'une étape intermédiaire de "spécifications détaillées des fonctions opérationnelles". Cet aspect sera développé ultérieurement, dans le cadre des améliorations apportées à la méthodologie.

3.5. PHASE DE PLANIFICATION

Cette deuxième phase, placée sous la responsabilité du fabricant du logiciel, a été de courte durée, car les travaux objets de ces trois étapes ont été naturellement abordés dès la première phase bien qu'ils n'en constituent pas l'activité principale. Il s'agissait d'en faire une synthèse et d'aboutir à des conclusions, formalisées dans une série de documents. Préalablement à toute réalisation, cette phase a donc consisté en un réexamen à vocation définitive de plusieurs éléments :

- a) la faisabilité du logiciel, notamment en ce qui concerne la taille mémoire et les charges de calcul, a été revue ; pour chacun des systèmes d'armes, il s'est avéré par la suite que les estimations faites à ce stade à partir des "spécifications détaillées du logiciel" avaient une bonne précision ($\pm 5\%$), mais que par contre elles étaient sur certains points notablement supérieures aux estimations initiales établies à partir des spécifications globales du système avionique.

Le matériel ayant été largement dimensionné dès sa conception, cette augmentation de taille mémoire des programmes n'a posé aucun problème et en particulier n'a pas eu d'influence sur les délais de livraison des calculateurs.

- b) Le recensement des besoins en personnel, matériel, logiciel nécessaires à la réalisation des programmes opérationnels doit être effectué à nouveau avec le maximum de rigueur au cours de cette étape. L'accent sera plus particulièrement mis sur les besoins en matériels dont les délais de réalisation sont les plus incompressibles. En particulier, c'est à ce stade qu'ont été décrites les spécifications de la "Baie de Validation du Logiciel" (BVL) dont l'importance fondamentale a déjà été mentionnée.
- c) Un réexamen approfondi des coûts et des plannings achève cette deuxième phase dite de "planification". En effet, ce n'est qu'une fois les étapes précédentes effectuées qu'il est possible de déterminer avec une certaine précision les coûts globaux (logiciels de base, matériel et logiciel de validation compris) et le planning de la phase de réalisation des logiciels opérationnels.

Nous avons assez bien su maîtriser nos délais : un retard peut se combler au moins partiellement par une augmentation de personnel dès qu'une organisation suffisamment structurée a été mise en place. Les plannings du logiciel opérationnel ont été établis en tenant compte de deux types d'impératifs :

- . besoins du maître d'oeuvre en matière d'intégration du système d'armes au sol et d'essais en vol,
- . date de mise à disposition des moyens de développement, de mise au point et de validation (ce sont ces dates qui ont constitué le chemin critique).

Ces contraintes ont été extrêmement sévères et nous ont conduit à effectuer des livraisons partielles de logiciel.

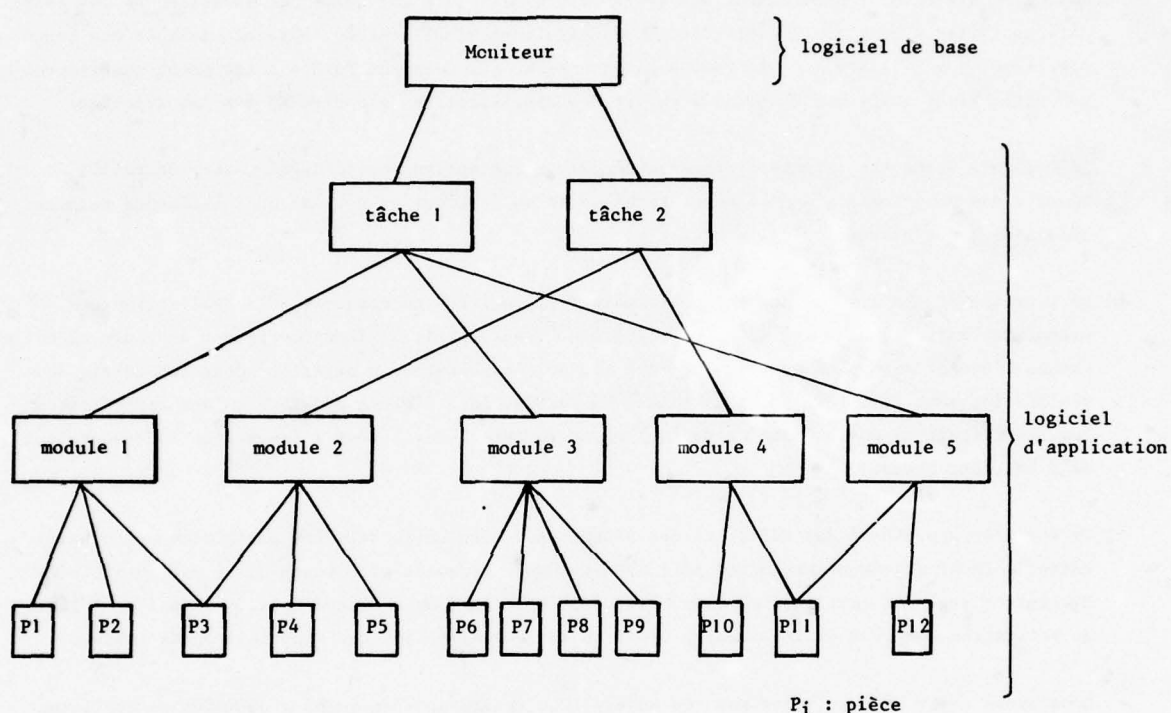
3.6. PHASE DE REALISATION

3.6.1. Structure du logiciel

La phase de réalisation est évidemment de la responsabilité entière du fabricant de logiciel. Le petit groupe d'ingénieurs qui a participé à la phase de définition (étape 1.2) a naturellement été conduit à jouer un rôle prédominant dans les deux premières étapes de cette troisième phase. Alors que la phase de définition précisait l'aspect opérationnel et la structure fonctionnelle du logiciel ("que faut-il faire ?"), l'analyse globale et l'analyse détaillée (étapes 3.1 et 3.2) en précisent l'aspect informatique et la structure organique ("comment réaliser ?").

Cette structure fait apparaître trois notions correspondant à trois niveaux hiérarchiques différents :

- tâche
- module
- pièce



Cette architecture modulaire présente une caractéristique fondamentale :

- tous les problèmes complexes à prédominance informatique et liés à la gestion du temps en général sont traités au niveau hiérarchique supérieur (tâche), les autres niveaux (module et pièce) les supposent résolus. Les tâches sont elles-mêmes exécutées sous le contrôle du "Moniteur Temps Réel", qui est un logiciel de base général, caractéristique du calculateur et non de l'application.
- tous les problèmes à prédominance fonctionnelle, c'est à dire la partie algorithmique proprement dite, qui constituent 95 % des spécifications du logiciel et sur lesquels 99 % des modifications ont porté, sont traités au niveau hiérarchique inférieur (pièce).

La pièce est l'unité élémentaire et fondamentale de la programmation : c'est le "composant" du logiciel, la brique de l'édifice ; sa taille est volontairement réduite, de façon à être facilement maîtrisable. Chaque pièce peut être considérée comme une boîte noire, dont les interfaces et les fonctions sont parfaitement définies ; par principe, chaque pièce est individuellement compilable, testable et archivable.

Un module est constitué par une suite d'instructions qui ne font qu'appeler des pièces ayant toutes les mêmes contraintes d'activation et dont le regroupement concourt à une seule fonction ; par exemple, un module ne fera appel qu'à des pièces s'exécutant à la fréquence de 50 Hz et relatives au guidage de l'avion. Certaines pièces particulières peuvent être appelées par

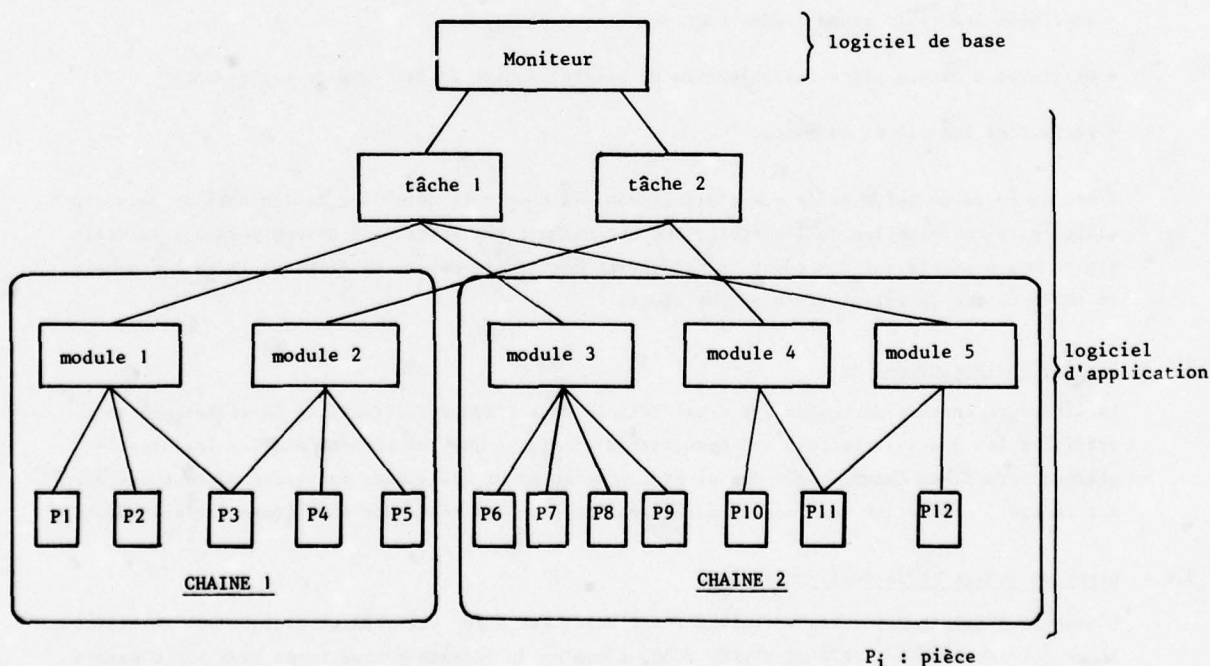
plusieurs modules. Si la pièce est le "composant" du logiciel, le module en est le "sous-ensemble". Chaque module a également été défini en tant que "boîte noire", avec ses entrées, ses sorties et ses propres jeux de tests.

Une tâche est constituée d'une suite d'appels à tous les modules ayant les mêmes contraintes d'activation, quel que soit leur rôle fonctionnel ; ainsi, tous les modules devant être exécutés à une fréquence de 5 Hz seront regroupés dans une seule tâche même s'ils remplissent des fonctions très différentes. Par ailleurs, c'est au niveau des tâches que sont également regroupées toutes les opérations d'entrée-sortie, de telle sorte que les modules, et plus précisément les pièces, ne travaillent que sur les informations présentes en mémoire, sans avoir à se préoccuper de leur transfert avec le milieu extérieur.

L'architecture du logiciel telle qu'elle est définie jusqu'ici, à son niveau hiérarchique le plus élevé ne rend pas compte du découpage fonctionnel et opérationnel du système d'armes : chaque fonction (localisation, désignation, etc...) est répartie entre les différentes tâches, une dizaine environ.

Pour remédier à cet inconvénient, il a été introduit la notion de "chaîne". Les chaînes sont au même niveau hiérarchique que les tâches ; elles consistent également en un regroupement de modules, mais effectué cette fois à partir de critères fonctionnels et non plus informatiques : l'ensemble des modules assurant la localisation de l'avion, quelle que soit leur condition d'activation (5 Hz, 50 Hz, etc...), constitue la "chaîne localisation".

Comme le montre le schéma ci-dessous, un module est l'élément commun entre une structure de logiciel établie selon des critères informatiques (qui aboutit à la notion de tâche) et une structure établie selon des critères fonctionnels (qui aboutit à la notion de chaîne).



Exemple de tâches :

- Travaux exécutés à la fréquence de 5 Hz,
- Travaux exécutés à la fréquence de 100 Hz,
- Travaux exécutés après une coupure d'alimentation,
- Travaux exécutés au démarrage du système,
- Travaux non prioritaires exécutés lorsqu'il reste du temps disponible (autotest, surveillance),
- etc...

Exemple de chaînes

- "Logique système" : assure le choix d'un mode de fonctionnement du système d'armes en fonction des commandes du pilote et de la disponibilité des équipements concernés.
- "Gestion des échanges" : assure le dialogue entre le calculateur et le reste du système.
- "Localisation" : calcule les paramètres instantanés déterminant l'avion par rapport au sol et à l'air en fonction des informations fournies par différents capteurs du système.
- "Désignation" : détermine et mémorise les coordonnées d'un point désigné par le pilote.
- "Guidage" : élabore les éléments nécessaires au guidage de l'avion vers une destination à partir d'informations issues des chaînes de localisation, de désignation et de postes de commande.
- etc...

Les spécifications détaillées de chaque chaîne sont définies dans l'étape 1.2.

3.6.2. Etape d'analyse globale

L'analyse globale (étape 3.1) consiste à :

- décomposer les chaînes en modules,
- identifier les diverses tâches par regroupement de tous les modules ayant les mêmes conditions d'activation,
- décomposer les modules en pièces,
- spécifier les traitements à effectuer par chaque pièce,
- attribuer à chaque pièce des objectifs de taille mémoire et de temps de traitement,
- identifier les pièces communes.

C'est au cours de cette étape que s'effectuent les compromis entre les contraintes de maintenabilité et d'optimisation du logiciel ; une attention toute particulière sera portée à la définition des éléments qui diminuent la modularité des programmes au bénéfice de la taille mémoire et de la charge de calcul (données partagées).

3.6.3. Etape d'analyse détaillée

La structure interne de chaque pièce est définie dans l'étape 3.2 (analyse détaillée) où sont précisées les données traitées en consultation et mise à jour et où sont établis les organigrammes détaillés. Chaque pièce devant être mise au point isolément, une fiche de test unitaire est rédigée, comportant des jeux d'essais statiques et les résultats théoriques correspondants.

3.6.4. Etape de codage et de tests unitaires

L'étape 3.3 (codage et tests unitaires des pièces) est assez triviale et quelque peu fastidieuse. En ce qui concerne le logiciel MIRAGE 2000, l'emploi du langage évolué temps réel LTR a permis une certaine diminution de la durée de cette étape.

3.6.5. Etape d'intégration et tests statique

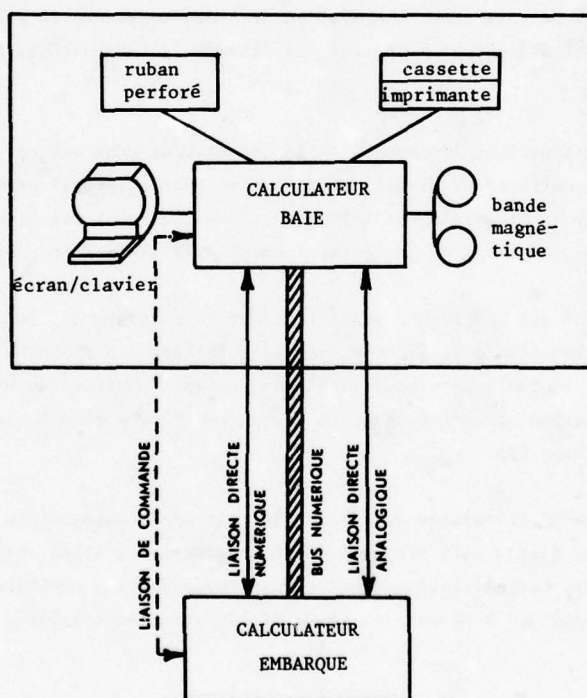
L'étape 3.4 (intégration et tests statiques) a pour objet de rassembler les pièces en modules et les modules en chaînes conformément à la structure définie par l'analyse globale. Il s'agit là encore de contrôles statiques, de même nature que les tests unitaires appliqués aux pièces ; ils ont pour but d'éliminer les erreurs d'interfaces entre pièces et entre modules. Ils sont

effectués conformément à des cahiers de tests rédigés préalablement pour chaque module et chacune des chaînes.

3.6.6. Etape de tests dynamiques et validation

L'étape 3.5 (tests dynamiques et validation) est fondamentale : il s'agit de tester dynamiquement les différentes chaînes, dont l'étape précédente a permis de s'assurer du bon fonctionnement statique, puis de les intégrer et de valider ainsi progressivement l'ensemble du logiciel opérationnel. Cette étape, dont le coût en heures ingénieurs a atteint pour chaque système d'armes environ le tiers de toute la phase de réalisation, s'est effectuée grâce aux outils de simulation et de mise au point particulièrement efficaces que constituent les BVL (Baies de Validation du Logiciel).

Comme le montre le schéma de principe ci-dessous, la BVL est conçue autour d'un calculateur qui simule les interfaces de tous les équipements dialoguant avec le calculateur principal dans le système d'armes. Cette simulation ne porte pas sur le fonctionnement complet et exhaustif de ces équipements, mais uniquement sur les fonctions nécessaires et suffisantes pour reconstituer l'environnement opérationnel du calculateur principal et de son logiciel.



Le calculateur de la baie est un calculateur EMD M182 identique aux calculateurs embarqués, ce qui a permis l'utilisation du même logiciel de base (système d'exploitation, moniteur, programme d'aide à la mise au point).

Les échanges d'informations entre les équipements simulés et le calculateur principal s'effectuent sur les liaisons opérationnelles du système d'armes (bus GINA, liaisons directes analogiques et éventuellement numériques) et dans les formes décrites par les spécifications détaillées du logiciel (les formats, les codages, la nature des signaux discrets sont respectés).

Le calculateur de la baie utilise deux sources d'informations :

- une bande magnétique contenant tous les paramètres d'un vol : accélérations, vitesses, position et angles d'attitude de l'avion, cap gyromagnétique, cap vrai, altitude radiosonde, altitude barométrique, Mach, distance et gisement du but, etc... etc... Cette bande est préalablement générée sur ordinateur universel par simulation en centre de calculs.
- un terminal écran/clavier permettant la représentation graphique des postes de commandes et de visualisation : grâce à un photostyle, un clavier de fonctions et un clavier alphanumérique, les actions du pilote et les pannes d'équipements peuvent être reproduites en temps réel.

Outre son rôle de simulation proprement dit, la BVL possède de nombreuses possibilités facilitant les tests dynamiques du logiciel : on peut bien entendu observer le comportement global du SNA sur le terminal écran, comme le ferait le pilote sur ses postes de visualisation, avec la restriction fondamentale que seules les informations traitées par le calculateur principal sont visualisées ; on peut également arrêter le calculateur à la fin de chaque cycle court, fonctionner en pas à pas, fonctionner en ralenti, visualiser les évolutions du contenu de toutes les mémoires, éditer ces valeurs sur imprimante, les sortir sur cassette à la demande ou de façon automatique ; on peut visualiser tous les échanges d'informations se déroulant sur le bus et les liaisons directes. Une liaison de commande assure le pilotage du calculateur embarqué par le calculateur baie et tous les échanges d'informations non opérationnels nécessaires aux tests.

Dans tous ces modes de fonctionnement, où le calculateur embarqué est volontairement ralenti, le temps réel est entièrement simulé de façon à respecter scrupuleusement le synchronisme du logiciel à valider : cet aspect est évidemment essentiel pour conserver aux tests dynamiques leur conformité temporelle avec le fonctionnement opérationnel du calculateur.

L'outil fondamental que représente une BVL permet de s'assurer de la conformité quasi-totale du logiciel d'application à ses spécifications ; l'étape 3.5 s'achève par une véritable recette usine du logiciel implanté dans son calculateur, avant livraison au maître d'oeuvre du système d'armes et intégration effective dans l'environnement opérationnel que constituent les équipements réels du banc SNA.

Sur un ensemble de 35 livraisons partielles ou complètes de logiciels, seulement une trentaine de non-conformités tout à fait mineures ont été détectées ; elles ont nécessité la modification d'à peine 150 mots. Cette fiabilité tout à fait remarquable a accéléré et considérablement facilité les travaux de la phase ultérieure (intégration du système d'armes au sol et essais en vol).

D'un point de vue méthodologique, les tests dynamiques ont été conduits de façon systématique et avec beaucoup de rigueur : un document décrivant avec précision tous les contrôles à effectuer dans un ordre chronologique imposé a été établi pour chaque chaîne de programme. Les résultats obtenus ont été consignés dans le même document.

3.7. PHASE D'EXPLOITATION ET DE MAINTENANCE

A l'issue de la recette usine, le logiciel est remis au maître d'oeuvre qui va pouvoir procéder progressivement à l'intégration du système d'armes. Une documentation accompagne chaque livraison ; elle précise en particulier les modifications de spécifications qui y sont incluses.

Le logiciel entre alors dans une nouvelle phase de vie comportant deux étapes : les essais au sol et les essais en vol. Ces deux étapes sont de l'entière responsabilité du maître d'oeuvre du système d'armes et, pour cette raison, ne relèvent pas de la méthodologie de production du logiciel.

En fait, ces essais en environnement opérationnel constituent une phase d'exploitation et de maintenance des programmes au cours de laquelle des adaptations d'importance diverse sont demandées. En effet, si le logiciel a été validé par rapport à ses spécifications détaillées lors des tests dynamiques sur BV, ces spécifications elles-mêmes ne sont validées définitivement par rapport aux besoins véritables qu'au cours de l'intégration du système ("A-t-on bien demandé ce qu'il fallait ?").

Ces adaptations induisent des retours sur les étapes précédentes selon les principes déjà mentionnés (paragraphe 3.2.g). Elles sont réalisées de la même manière que les modifications intervenant avant la livraison du logiciel et en particulier donnent lieu à tests dynamiques et validation.

L'exploitation et la maintenance des programmes ne constituent donc pas une phase particulière de la méthodologie mise en place mais se décomposent, de par leur nature même, en chacune des autres phases.

CHAPITRE IV

CONCLUSIONS

De façon générale, la méthodologie appliquée a donné satisfaction ; elle a permis d'atteindre l'objectif prioritaire : la qualité du logiciel livré au banc d'intégration du SNA. Cette qualité n'a pas été obtenue au prix de retards importants par rapport au planning ; le logiciel n'a pas constitué le chemin critique des systèmes d'armes.

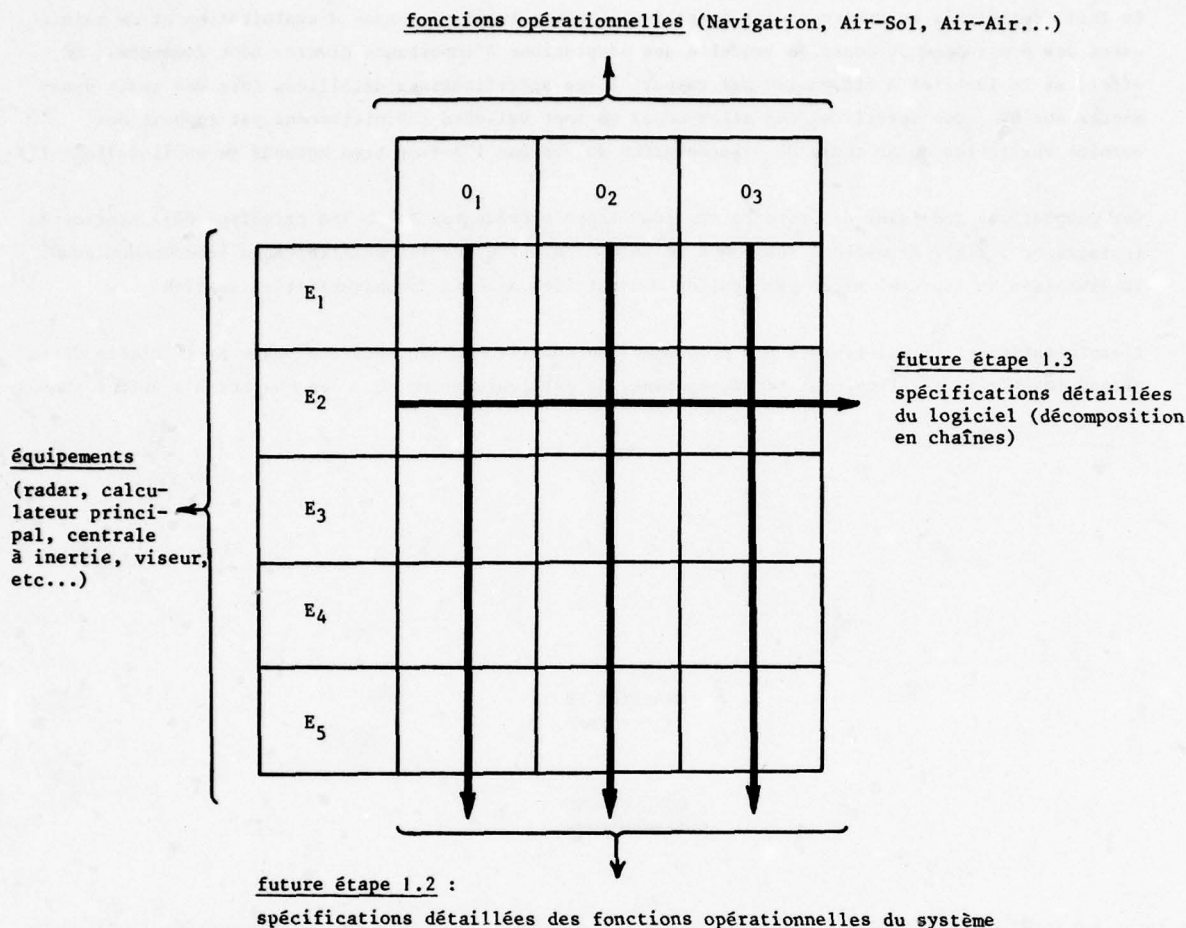
Cela ne veut pas dire pour autant que des améliorations ne sont pas à apporter à la structure mise en place. Elles concernent essentiellement la phase de définition ; un des objectifs à atteindre est de réduire le nombre d'adaptations apportées aux spécifications du logiciel pendant les phases de réalisation et d'exploitation : ces modifications sont en effet un des paramètres importants du coût final du logiciel.

Si l'expérience acquise est sans aucun doute un élément de progrès, il n'est pas question de s'en contenter.

L'élaboration des spécifications détaillées du logiciel (étape 1.2) s'est effectuée à partir d'un document définissant le système d'armes globalement sous son aspect opérationnel, dans une optique d'utilisation. A ce titre, les fonctions des différents équipements n'avaient pas nécessairement à y être décrites de façon très précise, opération par opération.

C'est précisément cette étape intermédiaire de "spécifications détaillées des fonctions opérationnelles" du système qui semble nécessaire avant d'entreprendre ce qui deviendrait la troisième étape, celle de la définition détaillée du logiciel.

S'appuyant sur les "spécifications globales du système" établies au cours de la première étape, cette seconde étape aurait pour objectifs de préciser dans le détail chacune des fonctions opérationnelles du système en y définissant le rôle et les interfaces de chaque équipement. On y procéderait opération par opération, toujours dans une optique "utilisateur" ; cela conduirait probablement à une excellente cohérence des traitements à l'intérieur d'une même fonction opérationnelle et à une structure de document plus compréhensible et mieux adaptée aux besoins des responsables de l'intégration du système sur le banc SNA : les fonctions logiciel du calculateur principal y apparaîtraient sous une forme plus opérationnelle.



L'étape 1.3 consisterait toujours en un travail de synthèse aboutissant à une décomposition fonctionnelle, en chaîne (cf. paragraphe 3.6.1), des spécifications détaillées du logiciel. L'aspect définition du système n'y serait pas nul puisque cette synthèse mettrait inévitablement en évidence certaines incohérences dans les interfaces entre fonctions opérationnelles, ce qui induirait des modifications au niveau de l'étape précédente, qui elles-mêmes pourraient avoir des conséquences au niveau de l'étape en cours.

Ce rebouclage entre une définition détaillée opérationnelle (étape 1.2) et une définition détaillée fonctionnelle (étape 1.3) devrait aboutir à des spécifications de logiciel moins évolutives en phase de réalisation.

Dans une perspective à moyen terme, nous nous intéressons par ailleurs aux différents travaux menés actuellement sur les langages de spécifications et les systèmes d'aide à la définition du logiciel.

Sur bien d'autres plans, des renseignements intéressants ont pu être tirés de ces expériences méthodologiques. Nous nous limiterons à évoquer ici deux points :

- a) LTR est un excellent langage évolué : la partie temps réel y est très complète. La partie algorithmique est bien orientée vers une programmation structurée, élément de fiabilité et de clarté ; elle permet de traiter en outre un grand nombre de types de données. Nous utiliserons de plus en plus ce langage pour nos applications futures.

Il ne faut cependant pas oublier que l'emploi de tout langage évolué exige de dimensionner le matériel en taille mémoire et puissance de calcul de façon à tenir compte du taux d'expansion du compilateur dont la maintenance doit par ailleurs être assurée par un personnel motivé et compétent.

Il ne faut pas non plus surestimer la réduction du coût direct de production apportée par un langage évolué ; ce gain n'est sensible que pour la partie "codage et tests unitaires" (étape 3.3) qui n'a représenté que 20 % du coût total de développement de nos logiciels.

Par contre, si une maintenance des programmes s'avère nécessaire sur une très longue période, la meilleure lisibilité des listings constituera un avantage certain, car ce sera en quelque sorte une assurance contre une qualité médiocre de la documentation.

- b) Dans le domaine avionique, le personnel logiciel doit être de haut niveau ; il doit pouvoir coopérer étroitement avec les concepteurs du système et les responsables des divers équipements. Un excellent profil est le jeune ingénieur à fort potentiel, de formation générale en électronique, automatisme ou aéronautique, avec une spécialité "logiciel" et une expérience pratique de programmation de quelques années.

Même avec ce type de personnel, des efforts constants doivent être consacrés à l'enseignement et à la bonne perception de la méthodologie. Celle-ci ne doit pas être ressentie comme une panoplie de règles pesantes et contraignantes, mais comme un ensemble harmonieux de directives, indispensables à la production d'un logiciel de qualité dans des conditions de délais et de coûts contrôlables.

La méthodologie ne doit pas être un frein à la créativité individuelle, au contraire elle doit constituer un moyen d'intégrer des hommes dans une structure au sein de laquelle ils se sentent plus efficaces.

EXPERIENCE IN PRODUCING SOFTWARE FOR THE GROUND STATION OF A REMOTELY PILOTED HELICOPTER SYSTEM

J. P. WEBBY
P. L. WESCOTT
M. I. TUCKER
H. M. SMITH

Westland Helicopters Ltd., Yeovil, Somerset, England

The production of a plan symmetric unmanned helicopter piloted remotely from a ground station required the setting up of a computer system within the ground station to control the aircraft, produce graphic displays and handle received data from the aircraft. This paper outlines the software system produced and describes the experience of the two contractors in meeting both cost and time schedules for the system using the so called Modular Approach to System Construction, Operation and Test (MASCOT) written in CORAL 66 language. The paper describes in some detail the overall design of the software and the methods used to test the complete software system in isolation from the total environment within which it was to be embedded.

1. Introduction

Westland Helicopters Limited (WHL) and Marconi Avionics (MAv) are jointly engaged in designing and developing a remotely piloted helicopter system to a requirement laid down by the United Kingdom Ministry of Defence.

The system consists of an aircraft and ground station. On board the aircraft is a sensor. The aircraft sends video signals and data to, and receives data from the ground station via an aerial and tracker. The aircraft is plan symmetric. In flight the aerial and tracker are automatically slaved in alignment. Within the ground station there is a pilot's station and an image interpreter's station. Figure 1 gives a diagrammatic representation of the system.

During the feasibility study it was decided to install a computer system within the ground station principally to:-

- (a) Shape the demands from the pilot's controls for onward transmission to the aircraft.
- (b) Monitor data from the aircraft to the ground station.
- (c) Display relevant information at the two work stations.

It may be noted that difficulties were encountered trying to implement these requirements using conventional hard-wired logic. In addition, the use of a digital computer gave the designer greater flexibility for changing control shaping functions and display formats.

Section 2 of this paper describes the function of the computer system in greater detail.

Section 3 describes the computer system architecture chosen to perform these functions.

Section 4 is a short note on system integrity.

Section 5, which represents the main bulk of the paper, describes how the computer software was designed, produced and tested by both WHL and MAv. Particular reference is made to the use of MASCOT (Modular Approach to Software Construction Operation and Test).

Section 6 draws some conclusions from the work.

2. Functions of the Ground Station Computer System

It was decided at an early stage in the project that, in order to deal with the complexity of the envisaged system in the way that would allow the maximum flexibility during development, some form of computational facility would be required in the Ground Station. The computer system as developed is to be used both for pre-flight ground station hardware checks and throughout the aircraft mission to control the vehicle, generate suitable displays and handle data received from the aircraft.

2.1 Pre-flight phase

The computer system is required to check its own status and to monitor the various peripherals connected to it for any malfunctions. This is achieved by loading the various software device handlers connected to a printing device for status and error reporting.

When these checks have been completed the software system is loaded and various mission planning and environmental parameters are entered. When these have been entered the computer system is ready for the flight phase to be entered.

2.2 Flight Phase

The various functions of the computer system during the aircraft mission are detailed below.

2.2.1 Control and Navigation

It is beyond the scope of this report to describe fully the philosophy adopted for the control and navigation of the aircraft. Such information may be found elsewhere (Ref. 1). However a brief outline is given here to give an appreciation of the complexity of the task required of the computer system.

The aircraft is required to operate in two distinct modes, each requiring conflicting methods of control. For take off and landing and observing areas of interest in remote locations a fine degree of control in order to achieve a stable hover is required. For navigating over long distances between areas of interest high speed and large changes in direction are required but with only coarse control. In order to reconcile these aims three flight modes can be engaged viz. a take off/landing mode, a navigation mode and a surveillance mode. Mode changes are requested by pressing buttons on the pilot's console and managed by the software system.

For the take off/landing mode the pilot has available a cyclic pitch control joystick for translational control of the aircraft, two trim wheels acting on the two control axes to facilitate 'hands-off' hovering and a collective pitch control slider for positioning in height. The aircraft will normally be in sight of the pilot during this mode and the axis of orientation of the joystick is the line of sight between the pilot and the take off point of the aircraft.

The surveillance mode allows the same controls but the authority of the joystick is limited in application by software. In this mode the pilot's control reference is the received video picture and the cyclic controls are thus oriented with respect to this view.

Navigation mode is selected when it is required to fly between areas of interest at relatively high speeds. The areas of interest are defined as a set of waypoints and selected using buttons on the pilots console. In this mode the pilot uses a speed control and heading control. These control demands are filtered through a first order lag and limited, the limit for the application of heading demand being dependent on the current speed demand. The lags are applied using a numerical integration routine, the time step for which is calculated by evaluating the elapsed time since the last pass of the code. This time step must be small enough for the response to be smooth and should be fairly constant throughout a mission. The pilots operation of the heading control is performed with reference to instrumentation, indicating bearing to fly to the next waypoint which is displayed on a CRT.

During a mission the aircraft may be required to fly at different heights. Since setting the collective control alone cannot maintain a fixed flying height the aircraft itself contains an automatic height hold system. The computer system is required to send the given flying heights to the aircraft lagging the application of these demands appropriately to attain a smooth response. The height demands are entered into the computer at the start of the mission and are applied as requested by button press on the pilot's console.

It should be stressed that the concept of differing flight modes is known only to the ground station computer, the aircraft itself receives the same set of commands throughout its mission. It is the responsibility of the software to resolve all the inputs into the appropriate format.

2.2.2 Data Monitoring

Data defining the status of the various aircraft parameters is transmitted from the aircraft to the ground station tracker. This data is smoothed and reduced by the tracker microprocessor system and sent on to the main processor which monitors all the information. Suitable messages are displayed on a CRT in the event that any of these parameters should become critical. Any breaks in transmission will be noted by the computer and again a warning will be displayed.

2.2.3 Displays

The computer system is required to drive two CRT displays, a plasma panel and X-Y plotter. There is a CRT at the pilot's station and image interpreter's station. The computer system is required to generate one of three display formats for the pilot and a fixed display for the image interpreter. Both operators may overlay the generated display with the sensor image.

The three display formats correspond to the three modes of flight. In the take-off/landing mode a range safety envelope and tracker limit are drawn with the superimposed position of the aircraft (denoted by a small square). In the navigation mode, suitable instrument-type displays are generated giving speed, course to fly to waypoint, present heading demand and heading control position. There are also indicators for the ground speed and range to waypoint. In the remote hover mode, a square boundary representing

a fixed ground coverage is shown which would usually be super-imposed with sensor image. All three display formats show the speed and position of the aircraft, currently selected waypoint and height demand.

2.2.4 Driving Peripheral equipment

The computer system is required to supply elevation angles to the tracker system, the tracker derives its own bearing and aircraft range from the locked-on signal from the aircraft's aerial. This information is then supplied to the computer along with the housekeeping data.

In order to monitor the progress of the aircraft during flight an X-Y plotter and plasma panel display draw the aircraft track from positional information supplied by the software at fixed intervals.

3. The Ground Station Computer Architecture

A Ferranti Argus 700S minicomputer was chosen to perform the central processing within the ground station.

This machine was to be augmented by:

- (a) an Intel 8080 microprocessor system to smooth the incoming data from the aircraft.
- (b) a microprocessor-based display system to draw the displays on the CRT. This device could store the three display formats referenced in section 2.2.3 thus minimising the load on the 700S and reducing picture change-over times.

The computer system architecture is shown in figure 1.

This paper concentrates on the development of software for the 700S. Two further 700S minicomputers were procured for software development; one to be sited at WHL, the other to be sited at MAV. Software was to be transported between these machines and the machine for the ground station, termed the target machines, using magnetic cassette tapes. The development machines included facilities for program development such as a teletype, line printer and CORAL 66 compiler that were not needed with the target machine.

4. System Integrity

The computer system is required to recognise, as far as is possible, any malfunction in hardware or software. Within the 700S minicomputer, a hardware malfunction is detected by Built-In Test Equipment procured from Ferranti. Software malfunctions are detected, wherever possible, by traps written into the software.

Should a malfunction be detected, an external relay is set causing pilots control commands to by-pass the computer system.

5. Software

Previous experience, both in the U.K. and overseas, of software-based systems for defence have not been entirely satisfactory, (ref. 2). Frequently, software development has exceeded time and cost estimates by wide margins and the end performance has been less than satisfactory. In addition, it has been found difficult to perform in-service modifications.

Many of these problems have been traced to inadequate control and management of the software. This in turn was due to an ill-disciplined approach to the software design and the fact that few if any standards had been imposed on those producing software. To resolve such problems, various techniques have been proposed (refs. 3, 4, 5).

One such technique is MASCOT (Modular Approach to Software Construction Operation and Test), which WHL and MAV have adopted for the project considered here. The definition of MASCOT is given in reference 6.

The next section explains how MASCOT has been used to develop the software for the Ferranti 700S minicomputer. It may be noted that certain features of MASCOT (e.g. features of a MASCOT evolutionary system) have not been used within the project to date.

5.1 Software Development using MASCOT

The life cycle of a software system can be considered as consisting of a number of stages from the issue of a requirement specification to in-service support.

This concentrates on three stages:

- (a) Overall design of the software system.
- (b) Implementation of that design. This stage consists of the detailed design of individual modules defined in the first stage and the coding of those modules.
- (c) Testing of the software.

It should be stressed that there was no clean boundary between these stages. For example, even after the overall design structure had been 'chilled', the design was changing in detail as a result of deficiencies found during stages (b) and (c) and changes in the functional requirement of the software.

In addition, due consideration was given at the earliest stage of this project of the way in which the software was to be tested.

5.1.1 Overall Design

Once the requirements of the software had been understood, the software design team created a network diagram describing the overall design of the software. Before describing this network diagram it should be stressed that the production of such a diagram represented a considerable amount of effort, particularly as WHL and MAV had no previous experience of MASCOT.

Figure 2 shows the network diagram of the overall structure of the software for this stage of the Supervisor project. In the paragraphs below, those words that have particular MASCOT meaning are underlined.

In Figure 2 the square boxes represent hardware peripheral to the 700S minicomputer. These are appropriately labelled. The arrowed lines represent flow of data. With a larger diagram these lines can be labelled with all the names of the data items being transmitted. The circular boxes are activities representing single processes which the software is required to perform. Conceptually all the activities are running in parallel. Data is passed between activities via channels (denoted I) and pools (denoted). The storage of data within a channel is inherently transient; a piece of data read from a channel is removed and cannot be read twice. A pool is used to store non-transient data (e.g. data in a table or directory). Reading an item of data from a pool does not remove it. The distinction between a channel and pool is made to aid the thinking of the designer.

MASCOT provides mechanisms, termed access procedures that allow an activity access to a channel or pool. Access procedures hide the detailed structure of a channel or pool from an activity and ensure an orderly and sustained flow of data between activities and between activities and peripheral hardware. In this implementation of MASCOT, data is passed from peripheral hardware to an activity and vice-versa, via a special channel termed a virtual interrupt and denoted .

A brief description of the function of each activity and the data flow is given below.

It can be noted that there is an overall flow of data from left to right.

The GSCS activity is responsible for the bulk of the control and navigation functions. Pilot's control demands are passed to it in a suitably ordered manner via a pool from the Control Monitor activity. Data describing the current position and state of the aircraft are passed to the GSCS activity via the Mission Control pool from the Data Distribution activity. The Command Assembler activity assembles the shaped control demands together with other data (e.g. undercarriage up/down) in a suitable format for transmission to the aircraft. The Display Manager activity has the task of generating and updating a number of different displays. The Watchdog Timer activity checks that the Data Distribution activity is being sent data sufficiently frequently, i.e. the radio link to the aircraft has not been lost.

Software used in the pre-flight phase for testing hardware consisted of either a suitably modified subset of the flight software described above or proprietary diagnostic software supplied by Ferranti.

The network diagram gave the design very good visibility and provided an effective focal point for discussions at design review meetings. It was also found that, fully labelled, the diagram could be appreciated by senior engineers with only scanty computing knowledge.

5.1.2 Implementation

The next stage involved designing and coding the software implied in the network diagram. The software was divided into three distinct parts:

- (a) The MASCOT 'kernel'.
- (b) WHL software
- (c) MAV software

As has been stated, the activities in the MASCOT network diagram are conceptually running in parallel. However, the Argus 700S has only a single central processor. This apparent contradiction is resolved by the fact that an activity will process a piece of code then wait for new data to become available via a channel, pool or virtual interrupt. Alternatively, it may wait for a specified time period. The MASCOT 'kernel' handles the orderly scheduling of the activities and the input/output. A software house was tasked with providing a MASCOT 'kernel' to the projects requirements.

The MASCOT network diagram divides the overall software into a number of functionally separate modules with well defined interfaces between them. WHL were tasked with programming those activities in the top half of the network diagram and MAV tasked with programming those in the bottom half.

The programming language CORAL 66 (Ref. 8) was used and found to be sufficiently powerful in all but a few isolated cases when in-line assembler code was used. CORAL 66 is the United Kingdom Ministry of Defence (MOD) standard programming language. By standardisation the MOD hope to significantly reduce software maintenance costs. The language was new to both the WHL and MAV software team. Its ability to incorporate structured programming principles probably led to code being produced more reliably than would have been the case if, say, Fortran IV had been used.

5.1.3 Testing

The software was tested 'bottom-up'. That is, initially individual activities were tested in isolation. Then certain groups of activities were linked together and this process expanded until finally the whole software system was tested. This is generally accepted as being the desirable approach to testing software (Ref. 4). The adoption of MASCOT made such an approach easy to implement.

Thus, initially, software was produced that is designed to test an individual activity. This test software loads relevant channels and/or pools with data and ensures the activity under test processes this data and outputs it to the required channels and/or pools correctly. An important consideration is that the activity must work correctly when operating on all the extreme values of the input data. It may be noted that activities connected to peripheral hardware via virtual interrupts can be tested independently of the peripheral hardware by programming the test software to stimulate the virtual interrupt.

Once individual activities had been rigorously tested, groups of activities were tested and the process continued until all the WHL activities and all MAV activities were working correctly together.

The complete software system is tested by embedding the software in a simulation of the ground station and aircraft such that the software is in an identical environment to its environment in operational use. Thus the overall software system can be tested by running missions in a realistic environment.

This system is outlined in Figure 3. Instead of the Command Assembler activity sending data to the transmitter, the data is sent to a PDP 11/55. Similarly, the PDP 11/55 sends data defining aircraft parameters to the Data Distribution activity. The PDP 11/55 is programmed to simulate the response of the aircraft to control demands. The response is only defined for flight dynamic parameters. Parameters such as engine temperatures assume a set of fixed in-range values unless a switch on the PDP 11/55 console is set when the selected parameter is sent to the Argus 700S out-of-range. The PDP 11/55 includes a graphic display showing the aircraft movement as viewed from a window in the ground station.

5.2 Timescales

The software system was fully tested, using the powerful simulation facility, within fourteen months of software design work commencing. This was achieved using a small team unfamiliar with either the Ferranti 700S, MASCOT or CORAL 66 (though specialist consultancy and guidance was provided by an MOD establishment).

The software design, coding and testing of individual activities took ten months. The remaining four months were needed to successfully integrate the WHL and MAV activities. This delay was not due to any significant deficiencies in the original software design but because the 700S minicomputer was initially incapable of running the activities with the required performance. However, by introducing one or two new data paths and by 'tuning' the priorities with which activities were scheduled, a satisfactory performance was eventually proved.

5.3 Quality

The software developed for the Supervisor ground station is directly responsible for the safety of the aircraft. As such it must be subject to quality control procedures.

WHL and MAV are implementing quality control procedures that the MOD define as mandatory for software for operational real-time computer based systems. The software design, coding and documentation is required to come up to approved standards (Refs. 9, 10). Adequate testing procedures must be seen to be defined which are capable of being run as and when personnel concerned with quality require. Strict configuration control

procedures relating to the storage, release, amendment and modification of software must be implemented.

6. Conclusions

This paper has been principally concerned with describing the approach used to design, code and test a relatively complex real-time computer system. This approach has been based upon MASCOT.

The main advantages of MASCOT have been

- it has enforced a well-disciplined, modular approach to the software design which allows quality control procedures to be readily implemented;
- the network diagram produced at the design stage has provided a description of the overall software structure that can be quickly and easily assimilated;
- it has provided mechanisms whereby real time software developed by different teams at different sites may be integrated successfully.

7. References

- 1) G. L. Hodge, J. Bartovsky. Control and Mission Planning of the Remotely Piloted Helicopter. Proceedings of the 3rd European Rotorcraft and Powered Lift Forum, Paper No. 36, September 1977.
- 2) Proceedings from the International Software Management Conference, AIAA, Spring 1977.
- 3) G. Ringland, A. R. Trice. Pilot Implementations of Reliable Systems. Software Practice and Experience, June 1978.
- 4) A Guide to the Management of Software-Based Systems for Defence. IECCA(p) 1/78 MOD(PE).
- 5) J. T. Shepherd. Avionic Software. Aeronautical Research Committee. ARC 37 385.
- 6) MASCOT Suppliers Association, The Official Definition of MASCOT. RSRE L-303(S). March 1978.
- 7) The ASWE Implementation of MASCOT for the Argus 700. ASWE WP-XBC-7702.
- 8) HMSO. The Official Definition of CORAL 66.
- 9) Specification for Air Technical Publications Covering Software for Operational Real Time Computer Based Systems. AvP 70, Spec. 4. August 1976.
- 10) Quality Assurance in the Programming and Use of Computers. QTM 16, August 1976.

8. Acknowledgements

Production of software for the R.P.H. ground station was a joint WHL, MAV exercise, funded by the Ministry of Defence (Procurement Executive). Acknowledgements are due to Mrs. M. H. Smith of WHL, and Messrs. A. Laker, D. Valentine and D. Morris of MAV. The software team is indebted to Mr. F. Albrow of the Royal Signals and Radar Establishment for considerable help and guidance in the development of the software.

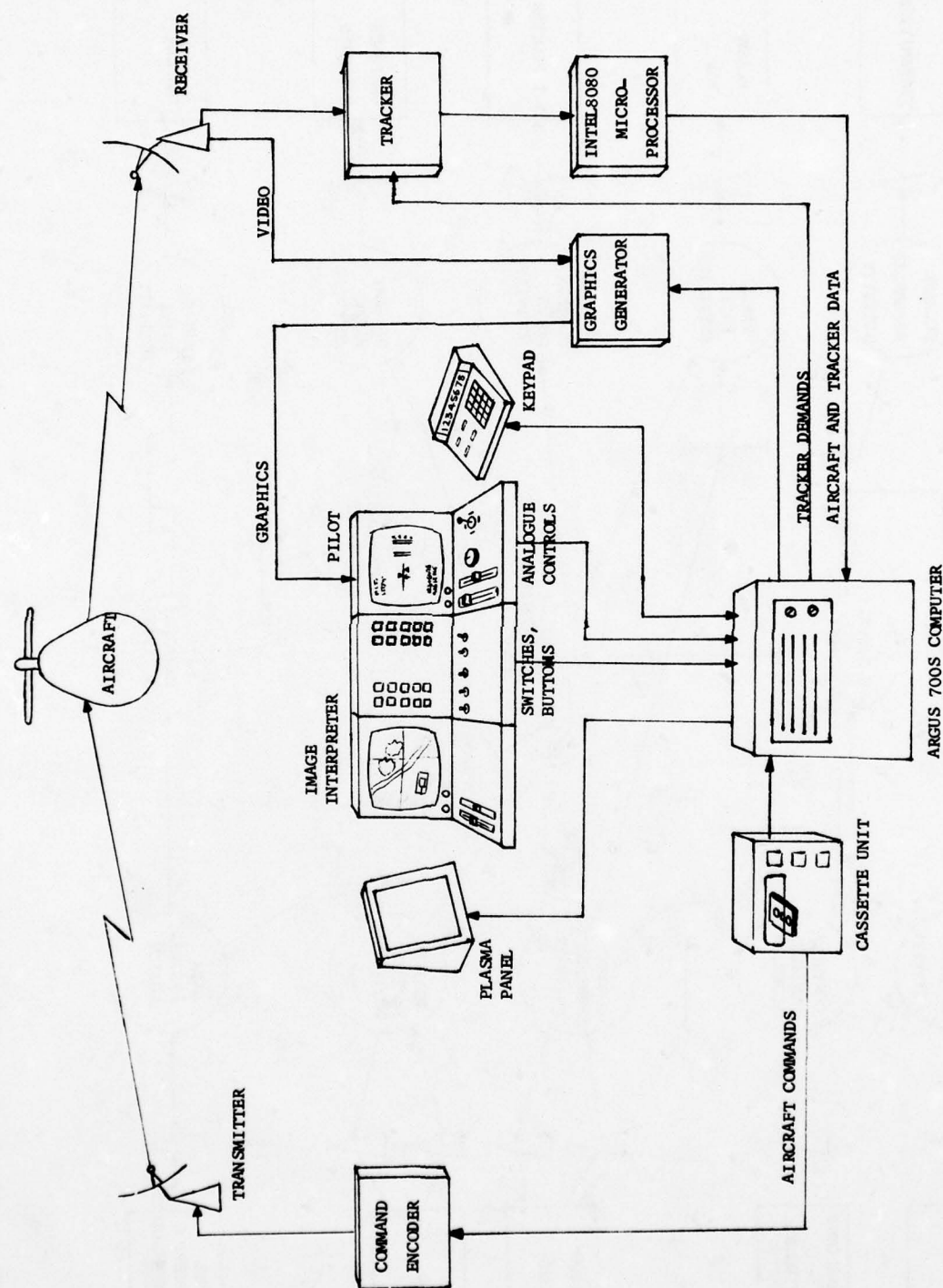


FIGURE 1 OUTLINE OF MINICOMPUTER SYSTEM IN RELATION TO THE GROUND STATION.

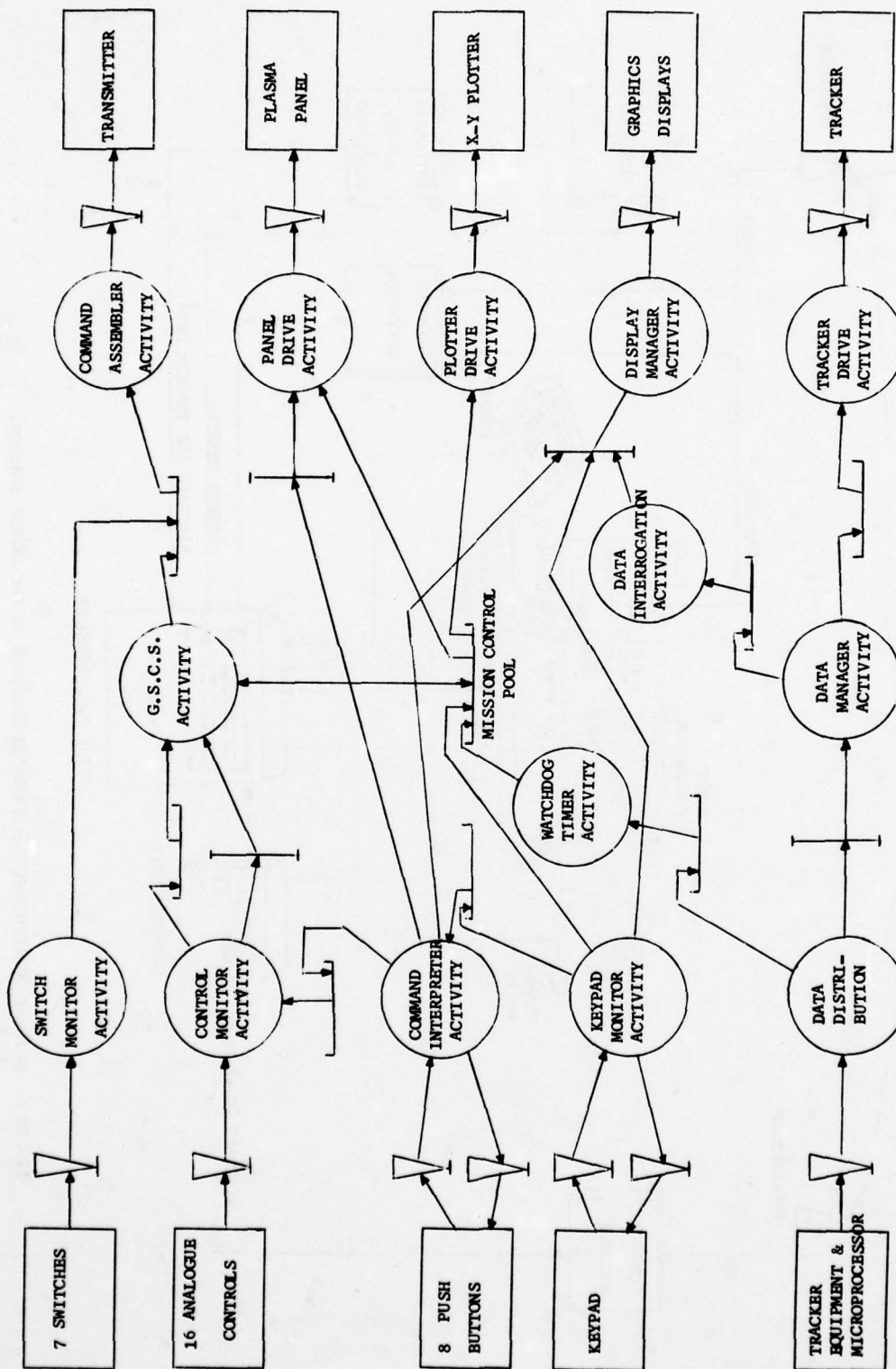


FIGURE 2 MASOT NETWORK DIAGRAM OF GROUND STATION SOFTWARE SYSTEM

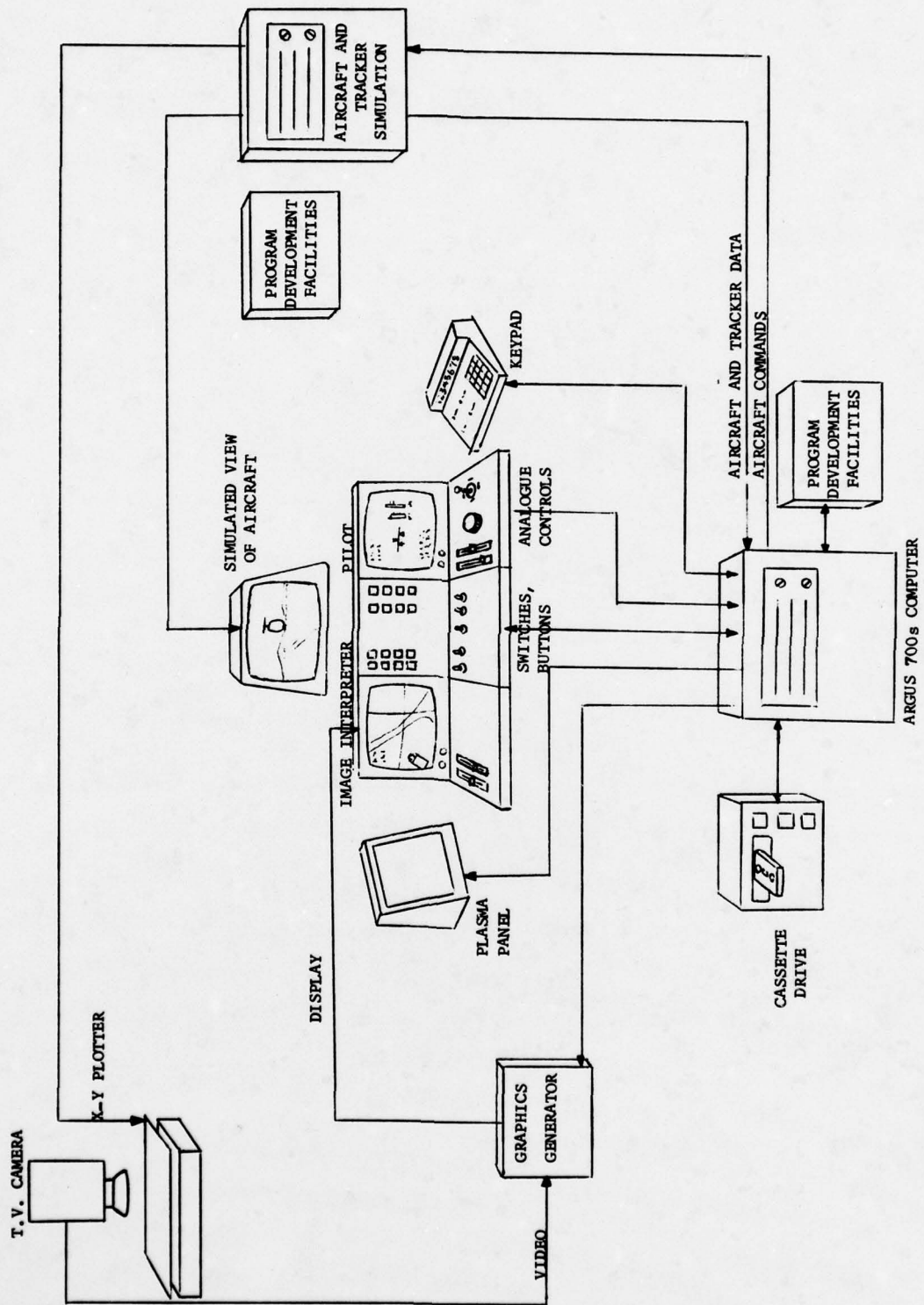


FIGURE 3 SCHEMATIC DIAGRAM OF GROUND STATION SOFTWARE TESTING SYSTEM.

SIMULATION USE IN THE DEVELOPMENT AND VALIDATION OF HiMAT FLIGHT SOFTWARE

Albert Myers
Aerospace Engineer
NASA Dryden Flight Research Center
P.O. Box 273
Edwards, California 93523
U.S.A.

SUMMARY

This paper describes the synthesis and use of real-time simulation in the development and validation of flight software for the highly maneuverable aircraft technology (HiMAT) remotely piloted research vehicle (RPRV). Four simulations were interfaced with varying amounts of actual flight hardware to produce dynamic system operation. How these simulation systems are used as a fundamental tool in the HiMAT development process is discussed.

INTRODUCTION

Under a joint program of the National Aeronautics and Space Administration (NASA) and the United States Air Force, two highly maneuverable aircraft technology (HiMAT) remotely piloted research vehicles (RPRV's) were built under contract and delivered to the NASA Dryden Flight Research Center (DFRC). These aircraft will be flown at DFRC to develop advanced technology applicable to fighter aircraft of the future. The HiMAT vehicle was designed to test advanced technology in the areas of aerodynamics, advanced composite and metallic structures, digital fly-by-wire controls, and digitally implemented integrated propulsion control systems (IPCS). An overview of the HiMAT RPRV program in terms of the various new technologies to be investigated is given in reference 1.

The HiMAT RPRV (fig. 1) is a 0.44-scale version of an envisioned full-scale fighter aircraft. The maneuverability goal for the RPRV is the ability to sustain an 8g turn at Mach 0.9 and an altitude of 7620 meters (25,000 feet). Figure 2 is a three-view diagram of the HiMAT RPRV with the control surfaces indicated. The canard flaps can be moved symmetrically for longitudinal control or asymmetrically for side force control; the ailerons and elevators are used for roll and pitch control, respectively; the elevons may be commanded asymmetrically for roll control or symmetrically for pitch control; and the rudders may be commanded collectively for yaw control or differentially as a speed brake.

The operational concept for the HiMAT RPRV is illustrated in figure 3. The 1530-kilogram (3370-pound) vehicle will be air launched from a B-52 aircraft at 13,700 meters (45,000 feet) and will carry 286 kilograms (630 pounds) of fuel for the J85-21 engine. The mission is to be flown under the control of a NASA research test pilot located in the ground-based RPRV facility cockpit. This facility has been used for flight test of other RPRV's at DFRC as described in reference 2. The flight test activity will be monitored on the ground by use of downlink telemetry. The vehicle will be equipped with landing skids for horizontal recovery on the Edwards dry lakebed.

All the flight control laws for both primary and backup operation will be implemented by the use of airborne and ground-based digital computers. This paper discusses the development and flight qualification of the software for these computers. Real-time simulation, involving varying amounts of flight hardware, has been a fundamental tool in the development and qualification process. Approximately 1800 hours of closed-loop simulation has been completed at DFRC in preparation for the first flight of the HiMAT RPRV.

HiMAT RPRV FLIGHT CONTROL SYSTEM

Primary Control Mode

The HiMAT RPRV control system operates in two principal modes, primary and backup. An overview of the HiMAT control system is shown in figure 4. In the primary mode of operation, aircraft sensor data are transmitted to the ground by means of a pulse code modulation (PCM) telemetry downlink. The downlinked information is used to drive the ground cockpit instruments and is input to the ground-based control law computer. Figure 5 is a photograph of the HiMAT ground cockpit, which contains a standard three-axis pilot input control system consisting of a stick, rudder pedals, and a throttle. Pilot commands are input to the control law computer by way of an analog-to-digital converter (ADC) interface. The control law computer combines the pilot input commands with the aircraft sensor data in the execution of the HiMAT control laws and formats a servo actuator command for each of the ten HiMAT vehicle control surfaces. These surface commands are output to the uplink encoder system and transmitted to the aircraft. The vehicle responses are then fed back to the ground-based RPRV system. Because the basic HiMAT aircraft has an extremely negative static margin, even a brief interruption of the augmentation provided by this system would cause the vehicle to depart longitudinally. The design considerations in the development of the HiMAT primary flight control laws are discussed in reference 3.

Backup Control Mode

Because control augmentation is critical to flight safety, a backup control system (BCS) is provided by the HiMAT onboard computer to fly the aircraft in the event the ground-based RPRV system is lost. The BCS, which can take control of the vehicle either automatically or by way of ground pilot command, is designed to recover the vehicle from any unusual attitude and cause the vehicle to orbit at a preselected altitude. Using a BCS control panel, a controller can issue drone-type commands to the HiMAT vehicle from either the ground cockpit or from a second panel in the back seat of the TF-104G chase aircraft. These commands include climb, dive, increase speed, decrease speed, turn left, and turn right. Using these commands, the controller can fly the HiMAT aircraft back to the Edwards dry lakebed and then select the BCS landing mode, which will provide automatic pitch control for landing while the pilot provides heading control.

HiMAT COMPUTER SYSTEMS

Ground-Based Computers

The ground-based RPRV facility control law computer, a Varian V-73, executes the HiMAT primary control system (PCS) software. The V-73 control law computer is a general purpose minicomputer with floating point hardware, writable control store, and 32,768 words of 330-nanosecond, 16-bit semiconductor memory. The V-73 computer has numerous peripherals which include a cathode ray tube (CRT) terminal, a magnetic tape drive, a line printer, a card reader, a magnetic disk storage system, and a paper tape reader/punch. In addition to these input/output (I/O) devices, the computer has a number of real-time data transfer devices, including an 8-channel ADC, a 16-channel digital-to-analog converter (DAC), 48 input discretes, 48 output discretes, a digital I/O bus interface with the V-77 telemetry decommutation computer, and a digital output interface with the dual uplink encoder system. These interfaces are depicted in figure 6.

In addition to executing the HiMAT PCS control laws, the V-73 computer is programmed to compute the cockpit display data that are not directly available from the telemetry downlink. From the downlinked static pressure, dynamic pressure, and free-stream air temperature data, the V-73 computer determines the pressure altitude, rate of climb, calibrated airspeed, Mach number, dynamic pressure, and the total vehicle energy derivative. In addition, the V-73 computer contains the ground-based portion of the HiMAT redundancy management and fault detection. This computer is also programmed to execute a preflight test sequence that automatically determines whether each of approximately 100 predetermined test results falls within expected tolerances. A complete summary of these test results, including a pass/fail indication for each test step, is printed at the end of the preflight test sequence.

With the exception of the I/O routines, the V-73 computer is programmed in Fortran IV. The V-73 software contains both real-time interrupt-driven code and non-real-time software. For the most part, system initialization is performed in the non-real-time portion with the actual flight control law software being executed in real time. Since the entire HiMAT flight software program cannot be contained simultaneously in memory, the program is overlaid—that is, a portion of the program is read into main memory from disk storage as required. No overlay activity takes place while the computer is operating in a real-time mode, however. The entire HiMAT V-73 flight software program contains approximately 9700 source statements of Fortran IV and 1800 statements of assembly language code.

The V-73 computer is interfaced with a Varian V-77 general purpose minicomputer containing 32,768 words of 660-nanosecond, 16-bit semiconductor memory. This computer contains no floating point hardware and is not as fast as the V-73 computer. The V-77 computer performs subframe PCM decommutation and passes the required downlink data to the V-73 computer. In addition, the V-77 computer decodes the information contained in seven downlink status words and drives the master caution warning panel which provides status information about the health of the various systems on board the HiMAT aircraft. The V-77 software, like that of the V-73, is primarily written in Fortran IV. The software is developed on the V-73 computer system, which is software compatible with the V-77 computer, and is cross-link loaded into V-77 memory. For this reason, the V-77 computer has only a minimal set of peripherals. The V-77 program consists of approximately 1700 lines of Fortran IV source code and 700 lines of assembly language code.

Airborne Computers

There are two computers contained in the HiMAT onboard computer system, one designated primary and the other backup. A HiMAT subcontractor designed and built the computers and provided the initial programming. The computational and memory capacities of the two computers are identical; only the I/O interfaces differ. The two computers communicate by way of a seven-word intercom buffer. The computers operate in parallel, asynchronously. In the event of a failure in either computer, the system is designed such that the alternate computer will operate in a degraded mode, allowing safe recovery of the flight vehicle. The principal functions of the primary computer are: (1) uplink processing, (2) downlink processing, (3) failure detection for the computers, sensors, servo actuators, and power systems (for both backup and primary flight control modes), and (4) degraded integrated propulsion control if the backup computer fails. The principal functions of the backup computer are: (1) uplink processing, (2) integrated propulsion control, and (3) backup flight control laws, active only if the aircraft is in the backup mode of operation. Each onboard computer contains 22,528 bytes (8 bits) of erasable programmable read only memory and 1024 bytes of random access memory. The software for the onboard computer is written entirely in Intel 8080 assembly language. The program for the primary computer contains approximately 19,000 lines of source code, and the backup computer program contains approximately 18,000 lines.

HiMAT SIMULATION SYSTEMS

The HiMAT simulation systems are made up of elements from various facilities at the NASA DFRF. The Cyber computer facility contains a Control Data Cyber 73-28 computer. The simulation facility contains V-73 and V-77 computers and a cockpit, all of which are functionally identical to the hardware existing in the RPRV facility. The RPRV facility contains the ground-based data link transmitters and receivers, V-73 and V-77 computers, and the cockpit from which the HiMAT vehicle is actually flown; hence, the equipment in this facility is construed as flight hardware. The HiMAT vehicle is an airborne facility and contains the onboard computer and airborne data link electronics.

There are four simulation systems used in support of the development and validation of the various HiMAT software driven systems. Each system has distinct characteristics that are necessary for the overall development and qualification of the flight software. In the order presented, each succeeding system uses more actual HiMAT hardware in the simulation than the previous one. Table 1 provides a summary of these simulations and lists some of the uses, advantages, and disadvantages of each. Each system requires a pilot to operate the controls in a cockpit. The HiMAT vehicle responses are computed as a result of pilot (control) inputs using the Cyber 73-28 computer. Electronic signals that represent these responses are fed back into the system so that closed-loop responses can be analyzed. The use of real-time simulation to detect any anomalous behavior of the flight software at the various stages of development and validation has been the key to the flight qualification of the HiMAT software.

In addition to these quantitative tests, several hours of piloted evaluation are conducted in each of the simulation configurations. These piloted sessions involve both general handling qualities assessments and the development and practice of the detailed flight plans.

The Cyber 73-28 is a large-scale computer which has been modified to run in a real-time mode under the control of a hardware real-time monitor and contains the following elements of real-time I/O: 64 DAC channels, 32 ADC channels, 128 input discrettes, 128 output discrettes, an output digital interface with a V-77 computer that simulates the PCM decommutation interface, and a digital input interface which simulates the uplink encoder system. Not all of this I/O capability is used in each of the four HiMAT system simulations described.

The Cyber computer, to compute the HiMAT vehicle dynamics, contains a complete nonlinear model of the HiMAT aerodynamics (including flexibility effects) that cover the entire HiMAT flight envelope. The HiMAT equations of motion are generally integrated at a 20-millisecond rate using a pseudo second-order Runge-Kutta integrator. Generation of the nonlinear force and moment coefficient terms is done at the integration interval rate by means of table lookup and linear interpolation. The most complex of the coefficient terms is the pitching moment, which is computed from the sum of 32 terms, each of which is a function of one to three variables. In each of the simulations, the Cyber computer is used to model the performance of the HiMAT IPCS system as a function of throttle command and flight condition.

All-Cyber Simulation

The HiMAT all-Cyber simulation is illustrated in figure 7. In this simulation, the HiMAT vehicle servo actuator system, the backup control system (which is executed by the onboard computer), the ground-based primary control system, and the uplink and downlink systems are all modeled, along with the HiMAT vehicle dynamics, using the Cyber 73-28 computer. This simulation is interfaced by way of the Cyber computer's real-time I/O system with the HiMAT simulation cockpit. Care is taken in this simulation to maintain strict code transportability between the PCS software running on the Cyber and that which will run on the V-73 control law computer; this can reasonably be done since both are programmed in Fortran IV. The BCS software implemented in the Cyber computer is programmed in Fortran IV from precisely the same detailed flowcharts that are used to produce the assembly language code which implements the BCS in the onboard computer. This simulation is the principal tool in the design and development of both the BCS and PCS software. Proposed designs can be thoroughly checked out in a dynamic environment before they are implemented in the computers to be used for flight. Because the Cyber is a fast, large-scale machine with high-speed peripherals, its use as the principal design and checkout computer greatly speeds up the development process. This simulation is the least complex to use from an operational standpoint and is the simulation most often used for pilot training and flight planning.

Cyber-Varian Simulation

The simulation facility V-77 computer is digitally interfaced with the Cyber 73-28 computer in such a way as to simulate the RPRV facility V-77 computer interfaced with the PCM decommutation system. The simulation facility V-73 computer is interfaced with the Cyber computer to effect a simulation of the RPRV facility uplink encoder system. These interfaces are such that the software for the Simulation Facility V-73 and V-77 computers is totally compatible with that for the RPRV facility computers. The HiMAT simulation cockpit is interfaced with the V-73 computer just as the corresponding V-73 computer is interfaced with the cockpit in the RPRV facility. This simulation configuration, which is shown in figure 8, allows much of the HiMAT PCS software to be validated in the simulation facility with a computer identical to that used in flight and with a full simulation of the HiMAT vehicle dynamics.

CASH Simulation

The computation and simulation of HiMAT (CASH) system goes a step further than the Cyber-Varian simulation just described. In this simulation, an actual HiMAT onboard computer is interfaced with the Cyber and simulation facility V-73 computers. This simulation is illustrated in figure 9. In this configuration, the V-73 computer is interfaced with an uplink encoder system which is hardlined to a decoder system (bypassing only the transmitter/receiver radio frequency link). The decoder system is then interfaced with the onboard computer, just as it would be in the flight configuration.

This simulation also contains a high fidelity electronic model of each of the HiMAT servo actuator channels. Each of the models is interfaced with the onboard computer, just as the onboard computer would be interfaced with the vehicle servo actuator electronics. The Cyber computer monitors these servo actuator models for the vehicle control surface positions and computes the corresponding vehicle response. In this configuration, the HiMAT BCS is executed in an actual flight computer. This simulation is a principal validation tool for the BCS software.

Iron Bird Simulation

Each of the simulations described previously will be active throughout the design, development, and flight test activity of the HiMAT program. Iron bird simulation, however, will be operational only during specific test phases. This simulation configuration (fig. 10) makes maximum use of actual flight hardware and, in fact, has the actual HiMAT RPRV flight vehicle in the loop. This configuration represents the most sophisticated of the HiMAT ground test phases. With the HiMAT vehicle in the hanger, the PCM downlink is hardlined to the RPRV facility, as is the uplink command system to the vehicle. All the vehicle control loops are active. The simulation of flight is accomplished by the addition of system interfaces with the Cyber 73-28 computer. Vehicle control surface positions, as measured by the servo actuator feedback linear variable differential transformers (LVDT's), are trunked to the Cyber computer. The simulated vehicle response and air data system outputs are trunked to the vehicle, summed with the actual vehicle transducer outputs, and input to the flight test instrumentation system PCM downlink system. In this configuration, the aircraft and RPRV facility systems can be made to work as if the HiMAT vehicle were in flight.

HIMAT FLIGHT SOFTWARE QUALIFICATION

All the HiMAT flight software undergoes two types of testing during the flight qualification process: verification testing and validation testing. Verification is the process by which it is determined whether the software performs exactly as specified. The verification process is accomplished by devising specific tests for each software task, conducting the test, and observing whether the task was accomplished according to specification. While verification testing may use one or more of the computer systems from a HiMAT simulation, much of it can be accomplished without simulating the dynamics of the vehicle. Validation is a broader task which seeks to determine whether the system, of which the software is a part, performs adequately to accomplish the flight requirements. Therefore, much of the validation testing requires simulation of the vehicle dynamics. The final stage of software qualification, using the iron bird simulation in ground testing of the full HiMAT RPRV system in a dynamic environment, allows identification and correction of system problems prior to flight.

Primary Control System Software Qualification

Once the preliminary version of the PCS control laws have been designed using linear discrete systems analysis, these control laws are coded in Fortran IV for evaluation in an engineering design version of the all-Cyber HiMAT simulation. Using this simulation as a major tool, the design and evaluation of the control laws are finalized. The PCS is then formally specified in a project document. The software is first implemented in a controlled version of the all-Cyber simulation. After initial verification is complete, the software is then programmed on the V-73 computer. Once the program is operational, the qualification process begins and the code is verified, element by element, to be in conformance with the specification.

Validation tests involve both quantitative and qualitative evaluations. For the validation of the first-flight HiMAT PCS software, 22 specific flight conditions were selected within the defined flight envelope. These conditions are listed in table 2. Table 3 summarizes 93 specific validation tests performed at these flight conditions. The complete sequence of validation tests is conducted on each of the four HiMAT simulation configurations. Any anomalies and any differences in test results from one configuration to the next must be accounted for and resolved as a part of the validation process. If, during the CASH or iron bird simulations, a discrepancy is found as a result of validation testing on the flight software or hardware, the problem is described in a written report and its resolution is tracked as a part of the HiMAT software configuration management system. Figure 11 shows the recorded time histories from the execution of a PCS validation test run using each of the HiMAT simulation configurations.

Backup Control System Software Qualification

The development and qualification procedure followed for the BCS flight software is similar to that used for the PCS. A developmental version of the BCS software is written in Fortran IV from a set of detailed flowcharts. As the BCS is developed, the detailed flowcharts become the software specifications for both the Fortran IV implementation in the all-Cyber simulation of the BCS and the Intel 8080 assembly language implementation in the HiMAT onboard computer.

The BCS software validation tests performed for first-flight qualification are shown in table 4. These validation tests cover two general areas: tests concerned with transfers from the PCS mode to the BCS mode, and those conducted entirely in the BCS mode. The entire sequence of 26 tests is performed using the all-Cyber, CASH, and iron bird HiMAT simulations. All anomalies and discrepancies between tests run in each configuration must be accounted for and resolved prior to completion of flight qualification. All discrepancies detected as part of the BCS validation tests are reported and tracked as discussed for the PCS testing.

CONCLUDING REMARKS

The highly maneuverable aircraft technology (HiMAT) remotely piloted research vehicle (RPRV) uses considerable sophisticated and complex real-time flight software. Several real-time simulation systems have been used by NASA to design and validate this software. These simulations have been developed in such a way as to enhance the efficiency of the software development process and to simultaneously minimize the time for which the RPRV facility and the HiMAT vehicle are required to be dedicated to the development and validation process. The use of real-time simulation to detect any anomalous behavior of the flight software at the various stages of development and validation has been the key to the flight qualification of the HiMAT software. The final stage of software qualification, using the iron bird simulation in ground testing of the full HiMAT RPRV system in a dynamic environment, allows identification and correction of system problems prior to flight.

REFERENCES

1. Lockenour, Jerry L.; and Layton, Garrison P.: RPRV Research Focus on HiMAT. Astronaut. & Aeronaut., Apr. 1976, pp. 36-41.
2. Edwards, John W.; and Deets, Dwain A.: Development of a Remote Digital Augmentation System and Application to a Remotely Piloted Research Vehicle. NASA TN D-7941, 1975.
3. Deets, Dwain A.; and Crother, Carl A.: Highly Maneuverable Aircraft Technology. Active Controls in Aircraft Design, P. R. Kurzhals, ed., AGARD-AG-234, Nov. 1978, pp. 6-1 to 6-14.

TABLE 1.—HIMAT SIMULATION CONFIGURATIONS

Configurations	Advantage	Disadvantage	Use
All Cyber	Best simulation for control law design. Least complex system (uses only one computer in realtime) Easiest system on which to develop software	Requires lengthy computation time Gives optimistic evaluation of BCS performance since vehicle sensor imperfections and resolutions are not modeled	PCS and BCS
Cyber-Varian	Best verification and validation tool for PCS software Uses identical V-73/V-77 system without impacting RPRV facility system	Does not operate in BCS mode Uplink and downlink system interfaces are simulated	PCS
CASH	Best validation tool for BCS software Best tool for validation of PCS-BCS interaction Greatest use of flight-identical hardware without impacting RPRV facility or HiMAT vehicle	Difficult system to use for control law or software design Highly complex system Downlink not simulated at full rate	PCS and BCS
Iron bird	Makes maximum use of actual flight hardware	Most complex system Requires dedicated use of RPRV facility and HiMAT RPRV	PCS and BCS

TABLE 2.—FIRST-FLIGHT STANDARD TEST CONDITIONS

Test condition	Mach number	Altitude, m (ft)
1	0.68	13,700 (45,000)
2	0.90	13,700 (45,000)
3	0.82	12,200 (40,000)
4	0.74	10,700 (35,000)
5	0.90	10,700 (35,000)
6	0.80	7,600 (25,000)
7	0.60	7,600 (25,000)
8	0.70	7,600 (25,000)
9	0.83	7,600 (25,000)
10	0.62	6,100 (20,000)
11	0.36	3,050 (10,000)
12	0.43	3,050 (10,000)
13	0.50	3,050 (10,000)
14	0.63	3,050 (10,000)
15	0.33	1,525 (5,000)
16	0.40	1,525 (5,000)
17	0.58	1,525 (5,000)
18	0.25	760 (2,500)
19	0.40	760 (2,500)
20	0.33	3,050 (10,000)
21	0.60	10,700 (35,000)
22	0.90	6,100 (20,000)

TABLE 3.—PCS VALIDATION TEST SEQUENCE

Test number	Mach number	Speed, knots	Altitude, m (ft)	Test conditions	Comments
1				Normal launch	
2	0.68	---	13,700 (45,000)	Launch with 0.9 m/sec (3 ft/sec) rms turbulence	Record separation profile from B-52 aircraft
3				Each surface offset $\pm 2'$ from launch condition	
4 to 26	22 standard test conditions			Nominal gains	Trim points and step responses, 2-second trim inputs
27 to 49	22 standard test conditions			Maximum gains	Step responses
50	---	250	1,525 (5,000)	Military power acceleration to 350 knots, afterburning acceleration to 350 knots	-----
51	---	250	3,050 (10,000)		
52	---	250	6,100 (20,000)		
53	---	250	9,150 (30,000)		
54	---	250	12,200 (40,000)	Speed brake (15-second) deceleration	Hold constant altitude
55	0.8	---	12,200 (40,000)		
56	0.8	---	9,150 (30,000)		
57	0.8	---	6,100 (20,000)		
58	0.7	---	3,050 (10,000)		
59	---	275	2,400 (8,000)		

TABLE 3. - Continued

Test number	Mach number	Speed, knots	Altitude, m (ft)	Test conditions	Comments
60 to 82	22 standard test conditons			Maximum aileron roll Full rudder step 2g to 3g turn Pull up to 10° angle of attack and release	-----
83	---	250	1,525 (5,000)	Gear extension	-----
84	---	250	915 (3,000)		
85	---	215	1,525 (5,000)		
86	---	215	915 (3,000)		
87	---	220	1,525 (5,000)	Pull up to 12° angle of attack	Gear down
88	---	220	1,525 (5,000)	Full aileron input to 90° bank angle	
89	---	220	1,525 (5,000)	Full rudder input	
90	---	200	915 (3,000)	Military acceleration to 350 knots and climb at 300 knots	Timed runs to 6,100 m (20,000 ft)
91	---	200	915 (3,000)	Military acceleration to 350 knots and climb at 350 knots	-----
92	0.8	---	9,150 (30,000)	3g turn	Three axis pulses
93	0.8	---	6,100 (20,000)		

TABLE 4. -BCS VALIDATION TEST SEQUENCE

Test number	Maneuver	Weight, kg (lbs)	Altitude, m (ft)	Mach number	Speed, knots	Comments
1	Launch Recovery ↓	1589 (3503)	13,700 (45,000)	0.68	180	-----
2		1589 (3503)	12,200 (40,000)	0.75	226	-----
3		1440 (3173)	7,600 (25,000)	0.70	292	-----
4		1440 (3173)	6,100 (20,000)	0.62	285	-----
5	Altitude hold ↓	1589 (3503)	10,700 (35,000)	0.80	272	-----
6		1440 (3173)	6,400 (21,000)	0.80	366	-----
7		1440 (3173)	5,800 (19,000)	---	300	-----
8		1381 (3043)	2,750 (9,000)	---	300	-----
9	Orbit ↓	1440 (3173)	7,600 (25,000)	0.80	338	Dive to orbit
10		1440 (3173)	7,600 (25,000)	0.80	338	Climb to orbit
11		1381 (3043)	---	---	300	Orbit at 3050 m (10,000 ft)
12		1381 (3043)	---	---	240	Orbit at 525 m (5,000 ft)
13	Turn ↓	1440 (3173)	7,600 (25,000)	0.80	338	Left and right turns
14		1381 (3043)	3,050 (10,000)	---	300	Left and right turns
15		1589 (3503)	3,050 (10,000)	---	300	-----
16		1381 (3043)	1,525 (5,000)	---	240	-----
17	Climb/dive ↓	1440 (3173)	9,150 (30,000)	0.80	304	Climb from 5800 m (19,000 ft) to 6400 m (21,000 ft), then dive to 5800 m (11,000 ft)
18		1440 (3173)	---	0.80	373	Dive from 5800 m (11,000 ft) to 2750 m (9,000 ft), then climb to 5800 m (11,000 ft)
19		1381 (3043)	---	---	300	-----
20		1381 (3043)	---	---	300	-----
21	Powered landing ↓	1589 (3503)	2,225 (7,300)	---	300	-----
22		1381 (3043)	2,225 (7,300)	---	300	-----
23		1381 (3043)	2,225 (7,300)	---	300	Abort at 15 m (50 ft)
24		1381 (3043)	2,225 (7,300)	---	300	Radar altimeter failure at 610 m (2,000 ft)
25	Engine-out glide Engine-out landing	1440 (3173)	6,100 (20,000)	---	285	Engine failure, transfer to BCS

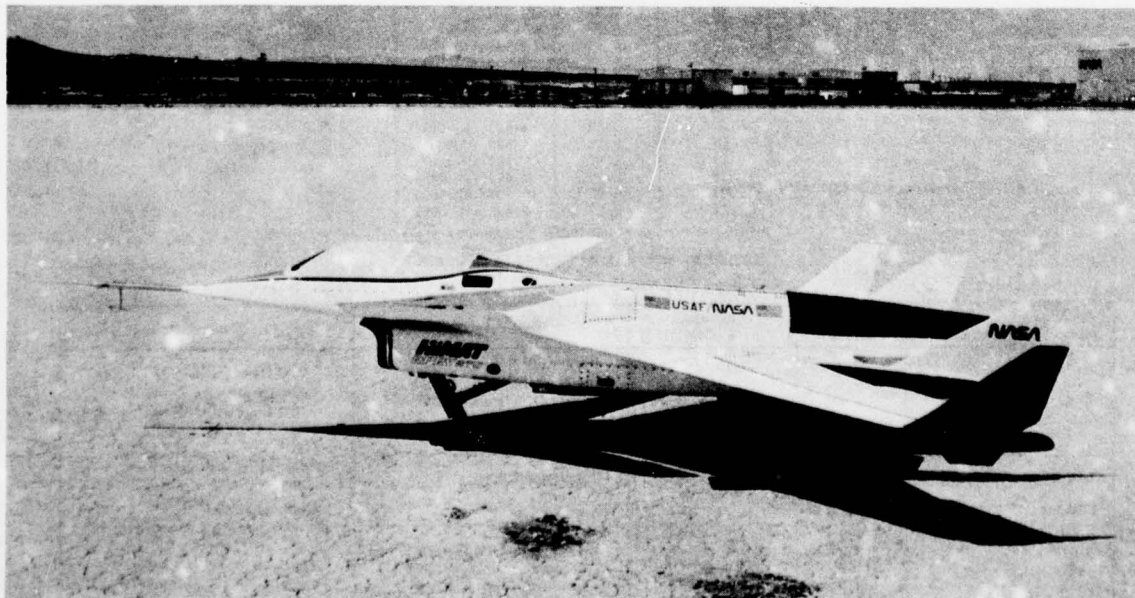


Figure 1. HiMAT RPRV on Edwards dry lakebed.

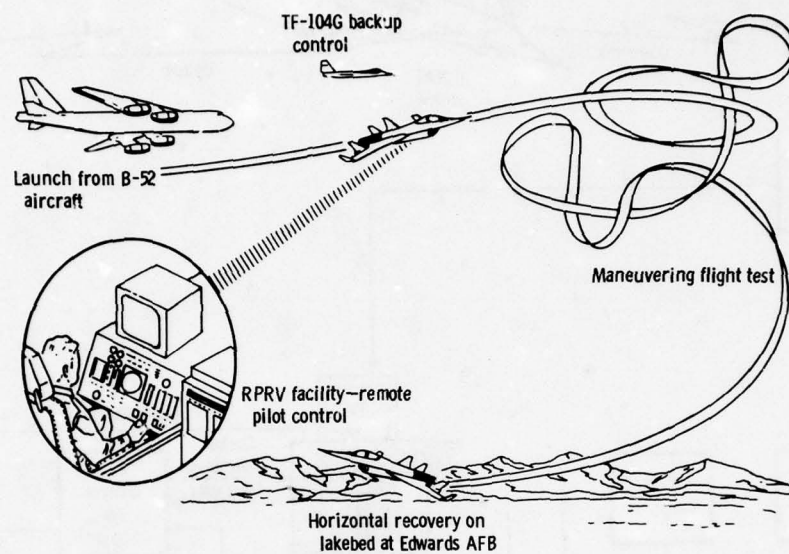


Figure 2. HiMAT RPRV.

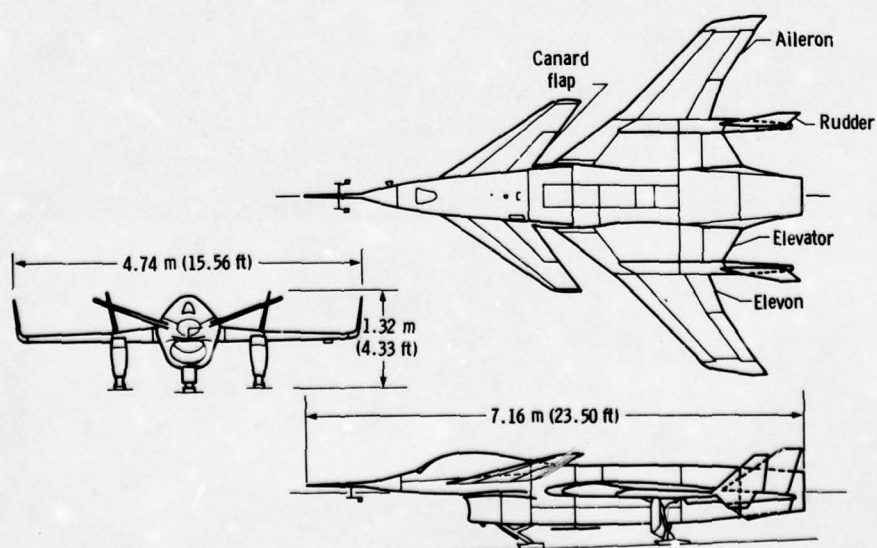


Figure 3. HiMAT operational concept.

AD-A076 146

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/6 17/7
ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQU--ETC(U)
AUG 79

UNCLASSIFIED

AGARD-CP-272

NL

4 OF 4

ADA
076146



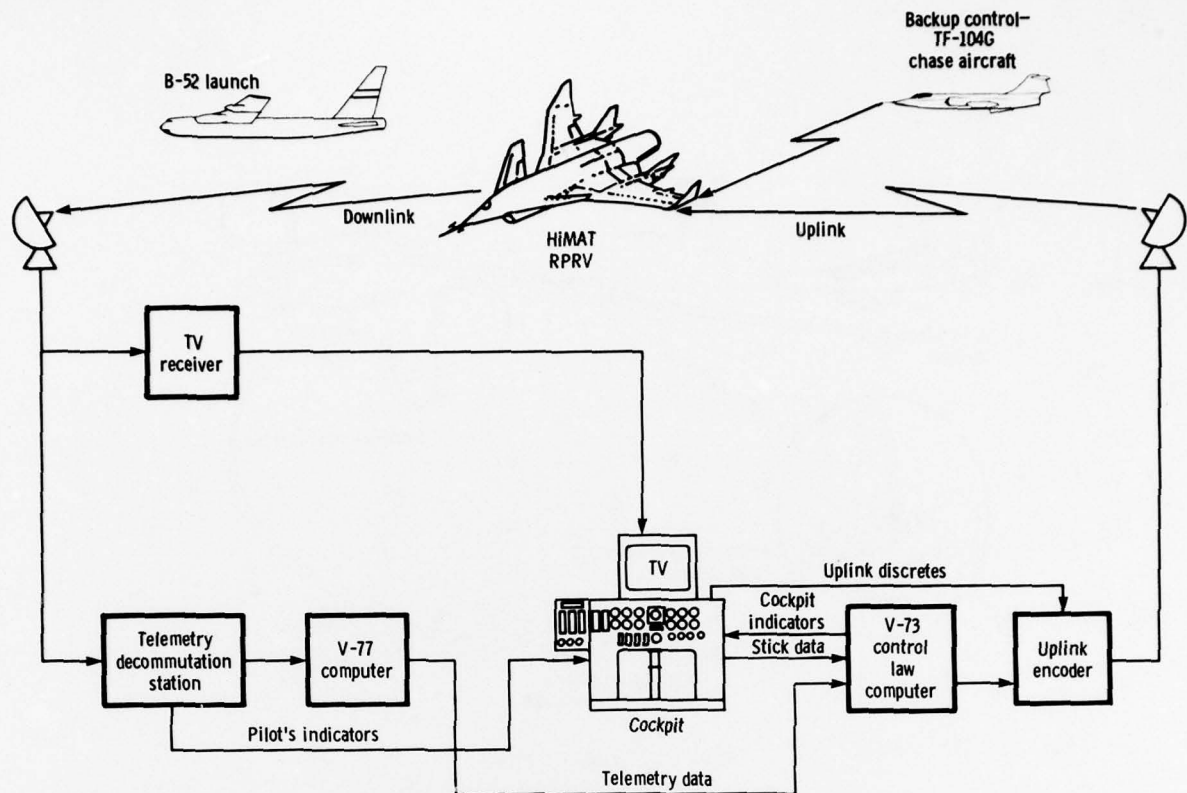


Figure 4. HiMAT RPRV control system.

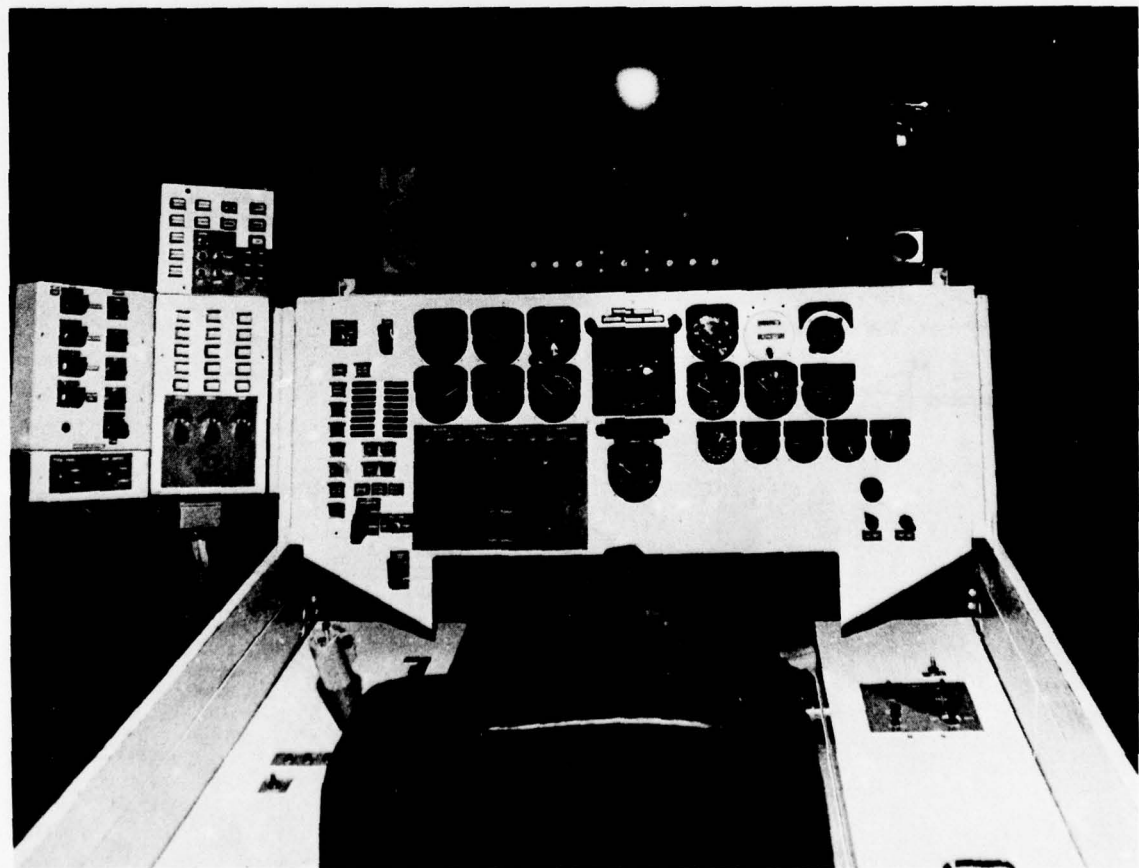


Figure 5. HiMAT RPRV ground cockpit.

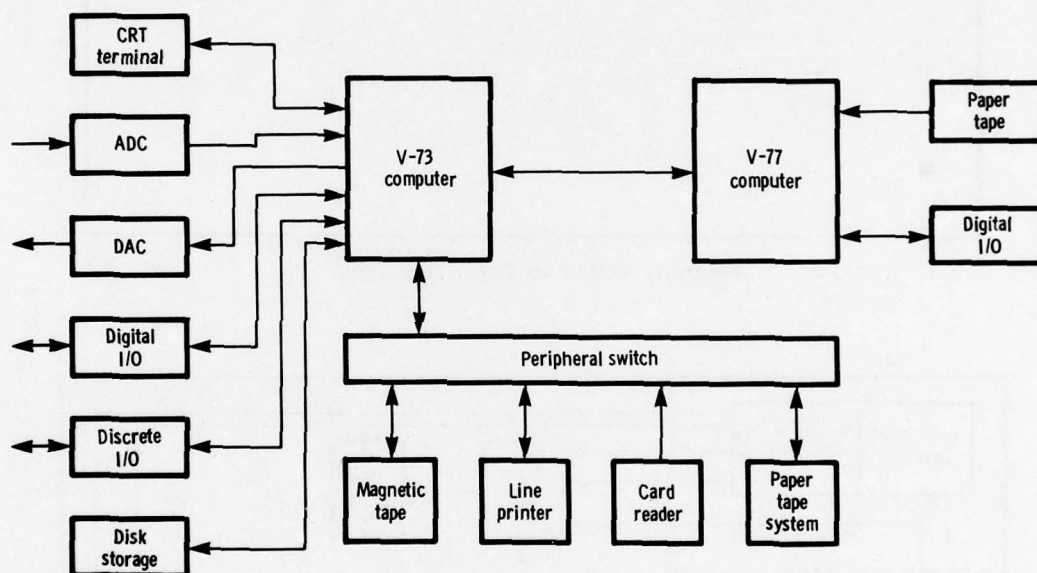


Figure 6. HiMAT RPRV ground computer system.

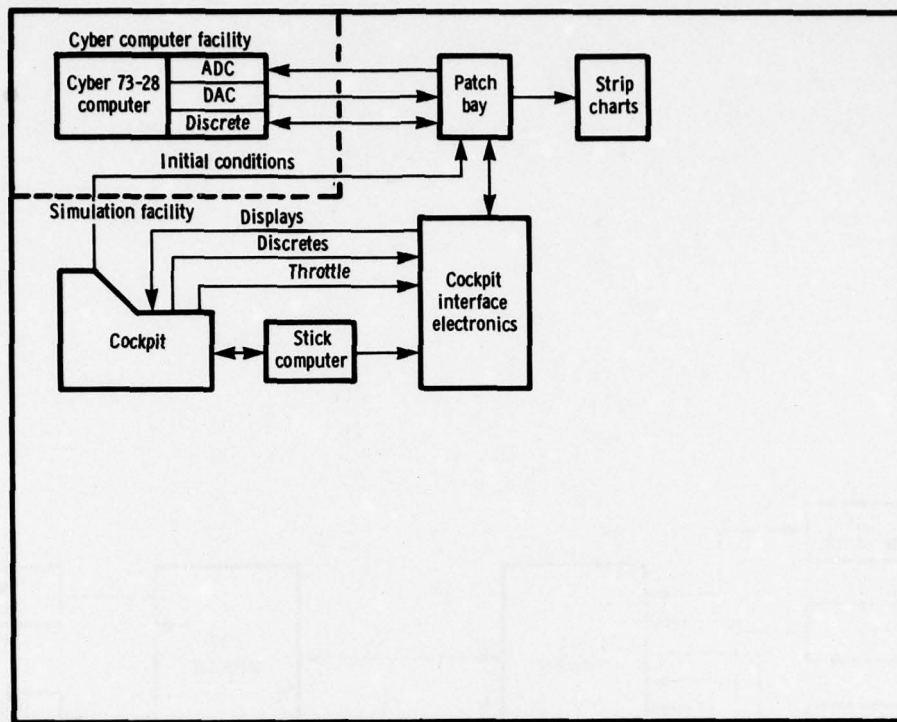


Figure 7. HiMAT all-Cyber simulation.

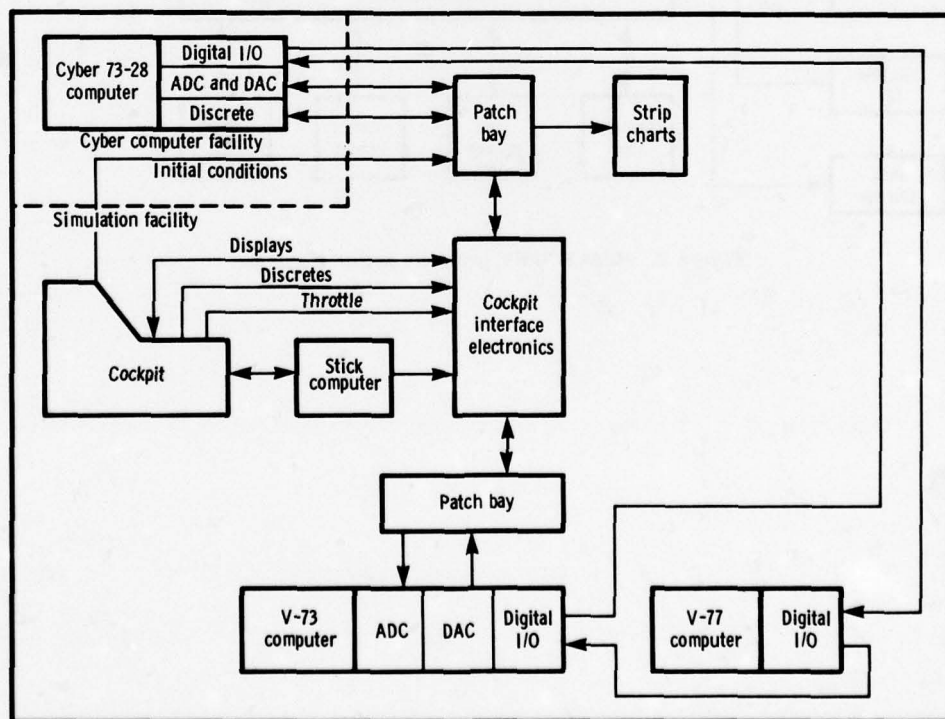


Figure 8. HiMAT Cyber-Varian simulation.

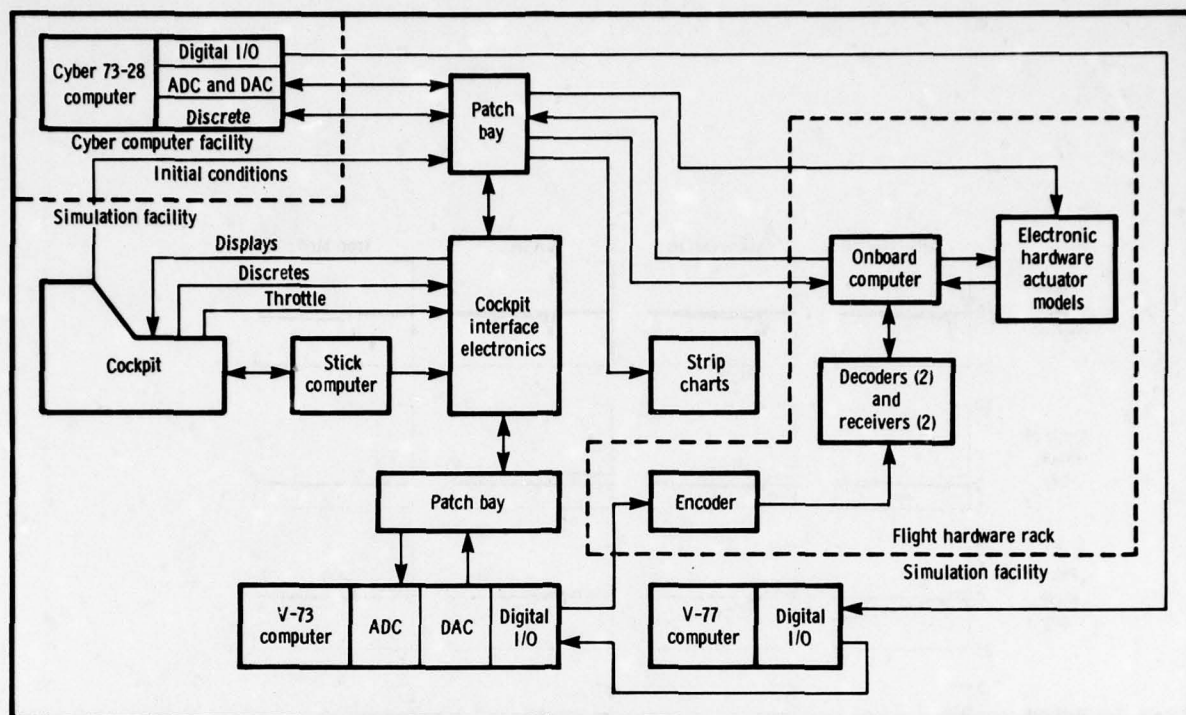


Figure 9. HiMAT CASH simulation.

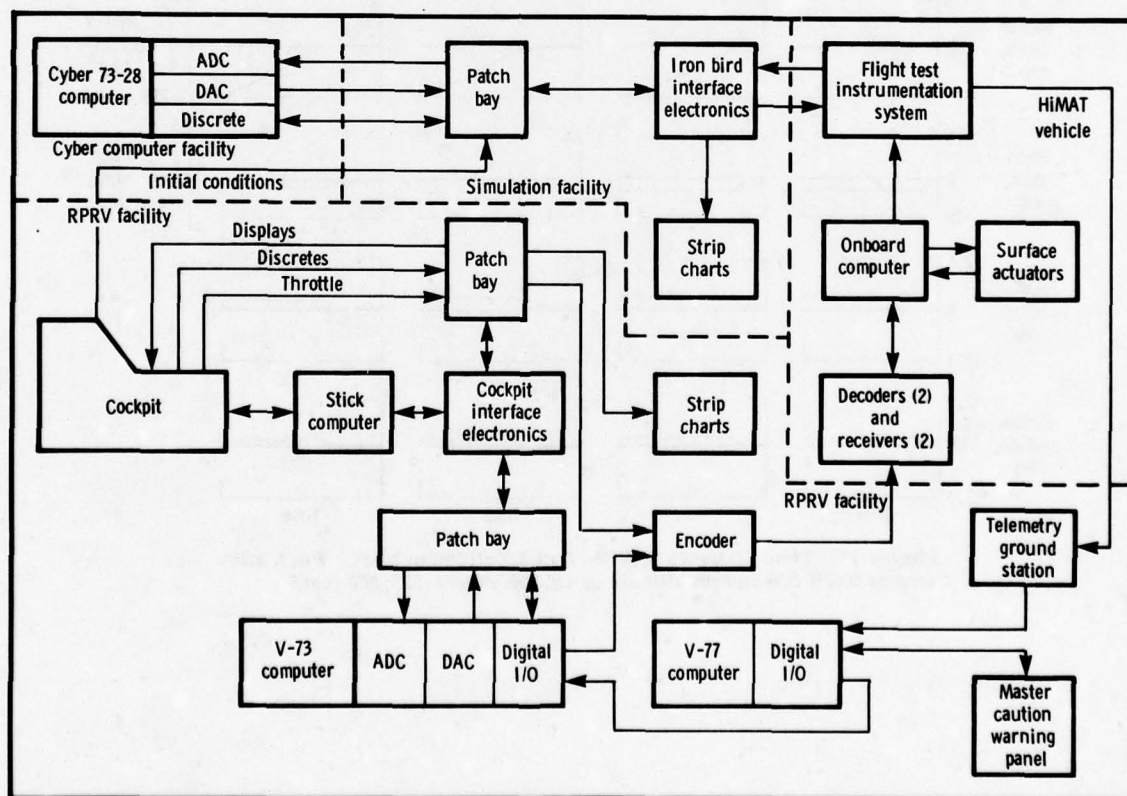


Figure 10. HiMAT iron bird simulation.

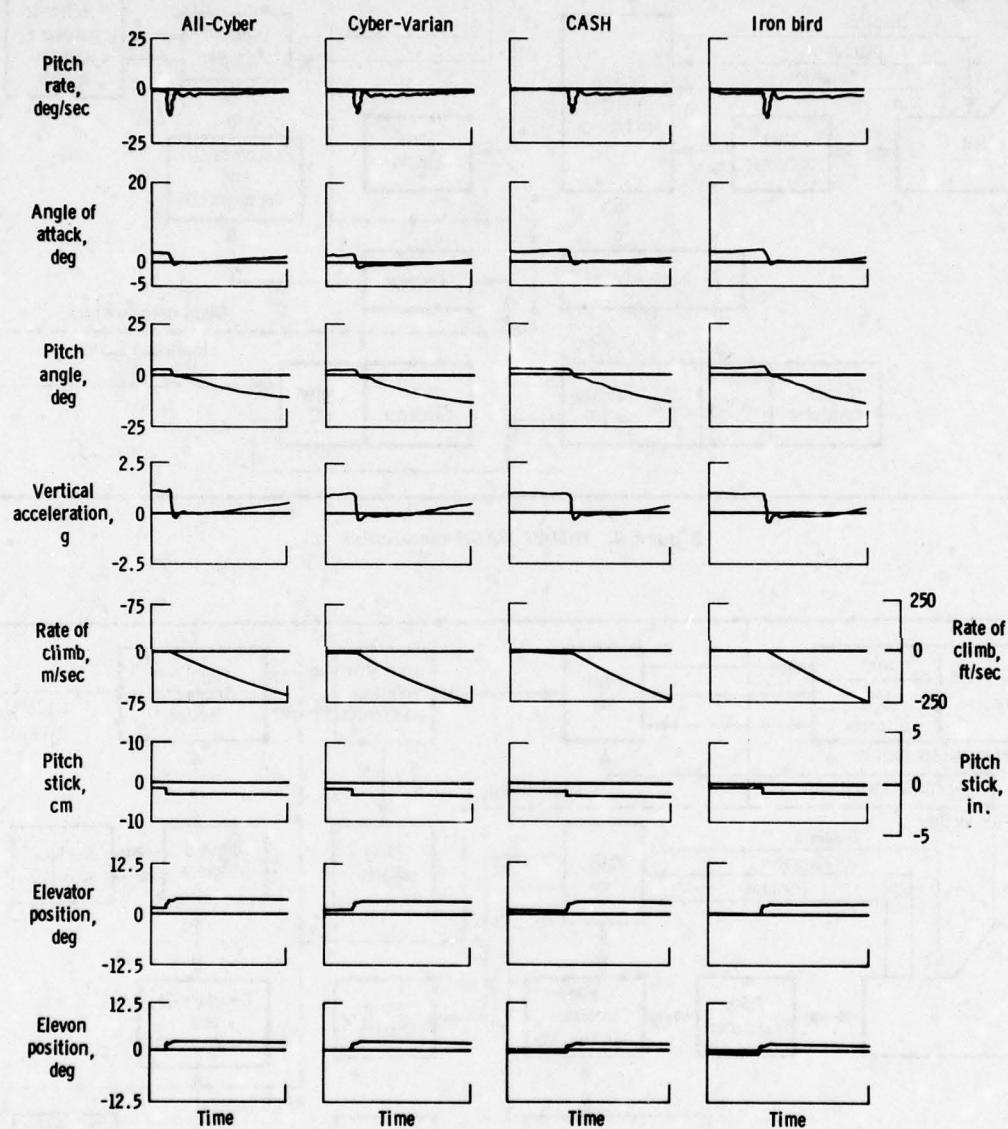


Figure 11. Time histories of HiMAT PCS validation tests. Pitch stick step at Mach 0.9 and an altitude of 10,700 meters (35,000 feet).

*FEDERATED MICROCOMPUTER SYSTEMS FOR ON-BOARD
MISSILE GUIDANCE AND CONTROL

by

Frank J. Langley
Principal Engineer
Raytheon Company (M3-14)
Missile Systems Division
Hartwell Road
Bedford, Mass 01730, USA

David S. Siegel
Office of Naval Research
Arlington, VA 20370, USA

L. Cdr Wayne F. Savage, USN
Director, Weapons Technology
Program
Office of Naval Research -
Code 212
800 N Quincy Street
Arlington, VA 22217, USA

Rollin E. Wehman, (NSWC)

ABSTRACT

This paper is based upon work performed under U. S. Navy contracts to analyze, structure and build modular digital missile guidance and control systems using standard industry microprocessors/microcomputers and associated memory and input-output interface components.

The types of missiles considered were air-to-air, ship-to-air and ship-to-ship, while the guidance and control functions analyzed were: target seeker signal processing, seeker head control and stabilization, state estimation, guidance laws, autopilots, ram-jet engine throttle control, warhead fuzing, telemetry and self-test.

Federated microcomputer systems were found to support hardware modularity at the system level, and a set of "macro-function" microcomputer modules provided the desired flexibility at the component level. Software modularity, although readily definable, has been traditionally difficult to maintain during the development process, particularly in single computer systems. Super-federation, by assigning one microcomputer per major functional algorithm, e. g., estimation, guidance etc., provides a means of enforcing software modularity through fixed, standard hardware interfaces.

A basic federated microcomputer system was built using standard-industry microcomputer modules integrated via high-speed programmable interface modules. The system constitutes the "hardware-in-the-loop", of a real-time missile simulation facility designed to evaluate the performance and flexibility features of next-generation, microcomputer based missile systems.

INTRODUCTION

Six years ago the Office of Naval Research became concerned with the inherent drawbacks of analog missile guidance and control (G&C) systems, namely: rigid/inflexible designs, limited performance and poor component commonality. To take advantage of the new trends in digital technology, e. g., microprocessors and digital signal processing, a study program was initiated to analyze the functional characteristics of the entire range of air-to-air missile G&C systems and determine the computer design requirements.¹⁻³ More recently, the Naval Surface Weapons Center initiated a similar follow-on study to address the requirements of multimission ship-to-ship and ship-to-air missiles, and furthermore, to fabricate a test-bed, federated microcomputer system. Throughout both studies, strong emphasis was placed on modularity, in hardware and software, since this was viewed as the key to achieving system functional flexibility at all stages in the life of any given missile system.

Compared to traditional analog control systems, digital implementations offer significant improvements in performance, functional capability and design flexibility through the use of sophisticated control algorithms as computer programs resident in read-only memory. Functions such as time variable estimation and guidance, adaptive autopilot control, fast Fourier transform signal processing and electronic counter-counter measures (ECCM) logic, while simple to execute digitally, cannot be performed easily or at all with analog techniques. Earlier applications of general purpose computers in missiles were limited by the relatively high cost, physical size and modest speed of mini-class machines using magnetic core memories, such that their role was typically confined to inertial reference support functions.

With the advent of low-cost microprocessors, system design emphasis can be shifted from achieving the ultimate in hardware economy to meeting more modular hardware and software design requirements.

STATEMENT OF THE PROBLEM

Despite the many functional advantages of digital versus analog systems, the simple substitution of a small general-purpose computer in place of the former analog circuits does not in itself solve all the problems encountered in the life cycle of a missile. The hard-core problems of advancing technology, changing threat situations, systems integration and logistics, together with the ever increasing cost of software, can result in an excessive premium being paid for digital missiles.

While throughput could be satisfied with a single, high performance, mini-class computer and a dedicated, special-purpose, target sensor signal processor, an unnecessary performance margin results in the case of low performance missiles, and form-factor and electrical interface problems arise across the range of missiles due to the many analog and digital discrete signals being converted and processed at a central point as opposed to being handled at the source. In addition, the design, assembly and checkout of major missile sections/functions, (e. g. seeker, warhead, flight control, telemetry), as completely operational modules is not possible with a single computer design approach.

*This work is sponsored by the U. S. Office of Naval Research and Naval Surface Weapons Center.

From a software viewpoint, programming complexity increases with program size and the time multiplexing of individual missile functions to meet the sampling and computational delay requirements of the various control loops, such that the modification or updating of any given function within the total program is fraught with virtually unknown and complex software interface problems, a situation which worsens as the level of coding diminishes. In other words, the interface problems cited for analog systems can reappear in digital missiles at the computer input-output interface and in a more devious manner within the invisible internal software. Software modularity supports the system flexibility requirement for changing threat/mission situations, but has proven difficult to achieve and maintain through a development cycle.

Medium range, point defense ship-to-ship and ship-to-air missiles are currently two distinct types, whereas a dual or multimission programmable guidance and control system could possibly provide a flexible "mix" of missiles to meet time-varying combat scenarios and threat conditions in naval engagements.

In terms of electrical interfaces between the missile and launcher and between missile subsystems, a serial digital multiplex bus interface has distinct merits in terms of standardization and simplicity, but the high cost and large physical parameters of traditional electrical drive circuits leaves room for improvement.

Within the domain of the microcomputer, no two microcomputer manufacturer's microbus interface schemes are the same, e.g. S-100 Bus, Intel MULTIBUS, National Microbus, etc., and similarly, the electrical interfaces of available support modules varies, e.g. analog-to-digital (A-D) and digital-to-analog (D-A) converters, memory modules, and serial digital interface modules.

Sensor signal processing throughput requirements exceed the performance of general-purpose microprocessors, such that special-purpose/dedicated processors are required involving approximately 150 standard-industry integrated circuits. As such the signal processor dwarfs today's microcomputer.

The above problems form some of the major issues addressed in the studies performed for the Navy and the solutions evolved are discussed in the following paragraphs.

MOTIVATIONS FOR FEDERATED SYSTEMS

The motivations for designing and building federated systems stem from the shortcomings of single computer systems and the availability of low-cost, large scale integrated (LSI) circuit microcomputers and associated I/O support circuits.

Hardware

Federated microcomputer systems simplify subsystem design, manufacture, interface, test and the inevitable modifications and updates. Figure 1 serves to illustrate the major differences between single and federated computer design approaches. In the case of the single computer system, a relatively large high-performance mini-type computer is subject to the varying form-factor constraints of a missile. To move the computer to a different location invariably entails the repackaging of hardware to fit the space available. A multiwire analog and digital interface problem also results from the concentration of data processing and conversion in one place.

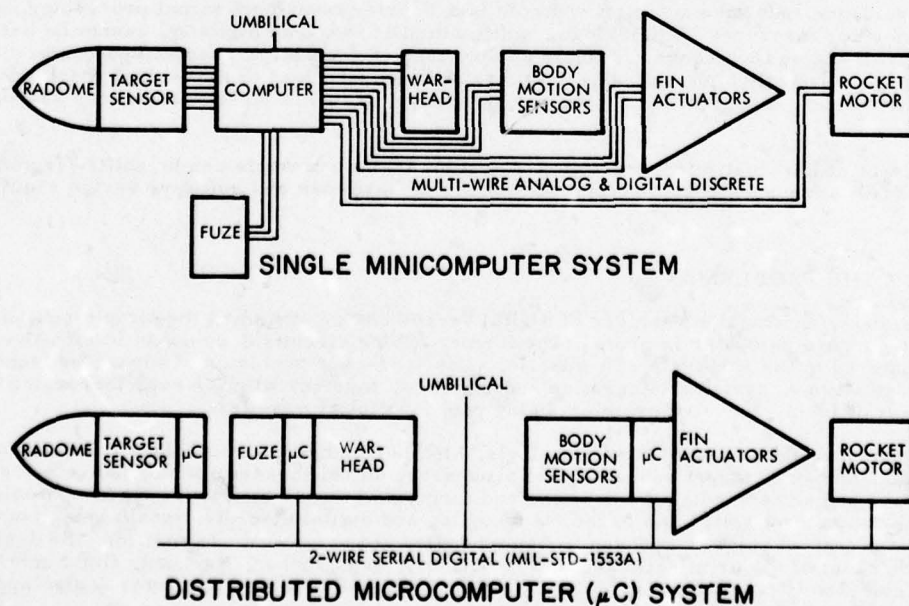


Figure 1 - Single versus Federated Missile Guidance and Control Systems

In contrast, the federated microcomputer system performs the data conversion and processing tasks at source, within each major subsystem, and allows a standard serial digital multiplex interface to be used between subsystems and the launcher. This partitioning is discussed at greater length in subsequent sections of this paper.

Software

The merits of modular structured software, although well-appreciated in this day and age, are somewhat idealistic and difficult to achieve and maintain. Figure 2 illustrates a rational, modular, hierarchical control structure for a single computer missile guidance and control system. All calls are made downward from the executive to subordinate mode supervisors and supporting functional program modules. However, under the normal pressure of tight development schedules the finished software is

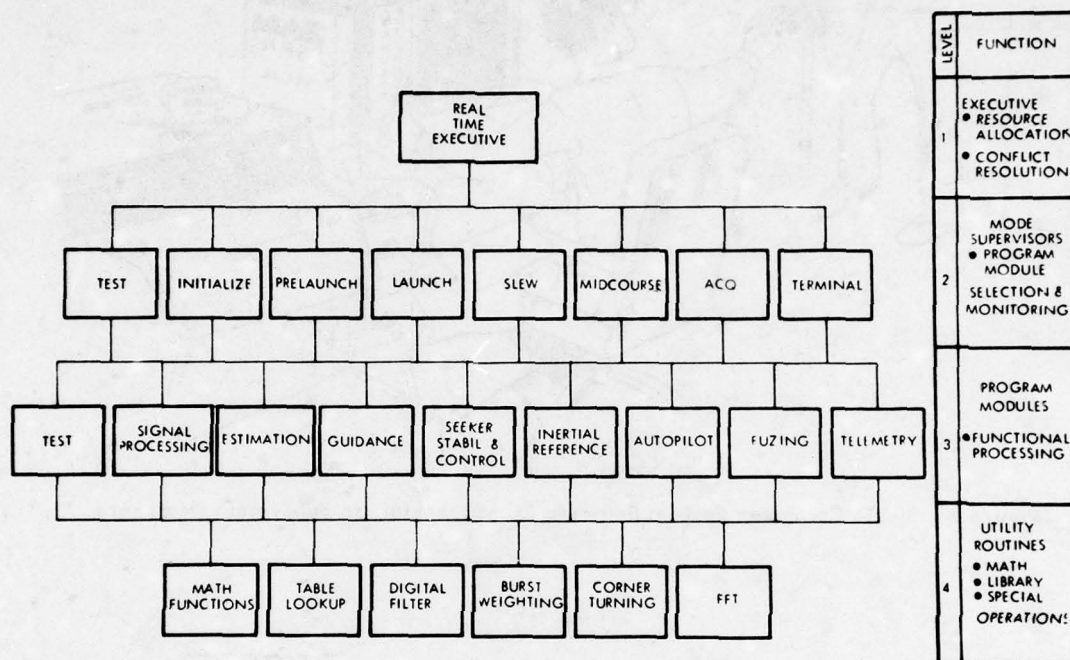


Figure 2 - Modular Hierarchical Software Control Structure for Single Computer Missile Guidance and Control System

subject to short cuts which invariably violate the original clean modular lines of the control structure. The outcome of a degradation in software modularity is realized more in the later phases of the development process when changes and substitutions are required, (Figure 3). Since software costs are pegged to ever-increasing labor rates, the impact of the deficiencies in the design structure together with their significant factors outlined below are not apparent until the total cost of the finished product is paid.

$$\text{Cost Per Instruction} = \frac{\text{Design Cost} + \text{Coding Cost} + \text{Verification Cost} + \text{Maintenance Cost}}{\text{No. Lines of Code}}$$

Determining factors:

- 1) Predominantly labor costs, dependent upon:
 - a) Firmness of Requirements
 - b) Proportion New versus Proven Algorithms
 - c) Size
 - d) Complexity
- 2) No. lines of code, dependent upon:
 - a) No. Functions Assigned to Software
 - b) Level of Programming Language

Experienced software costs over the past few years indicate an average cost of \$40 to \$60 per instruction for a fully commissioned system in terms of new, real-time, operational programs; and between \$8 to \$30 per instruction for more standard routines. Whereas the cost of semiconductor memory is reckoned to be in the order of millicents per bit, a 50-word subroutine typically costs \$3000 as a finished product. Microcomputer hardware, on the other hand, enjoys a volume market with modules selling in the tens of dollars. This situation emphasizes the need to be able to reuse or recycle program modules and to curb the tendency of designers to "do it in software" when in doubt about the requirements of a specific system function.

Throughput

Studies have shown that the throughput requirements for single computer systems can reach the two million operations per record (two Mops) mark, (Figure 4). Whilst a machine could be designed and built to meet the speed requirement, the tendency has been to add more functions: during the initial

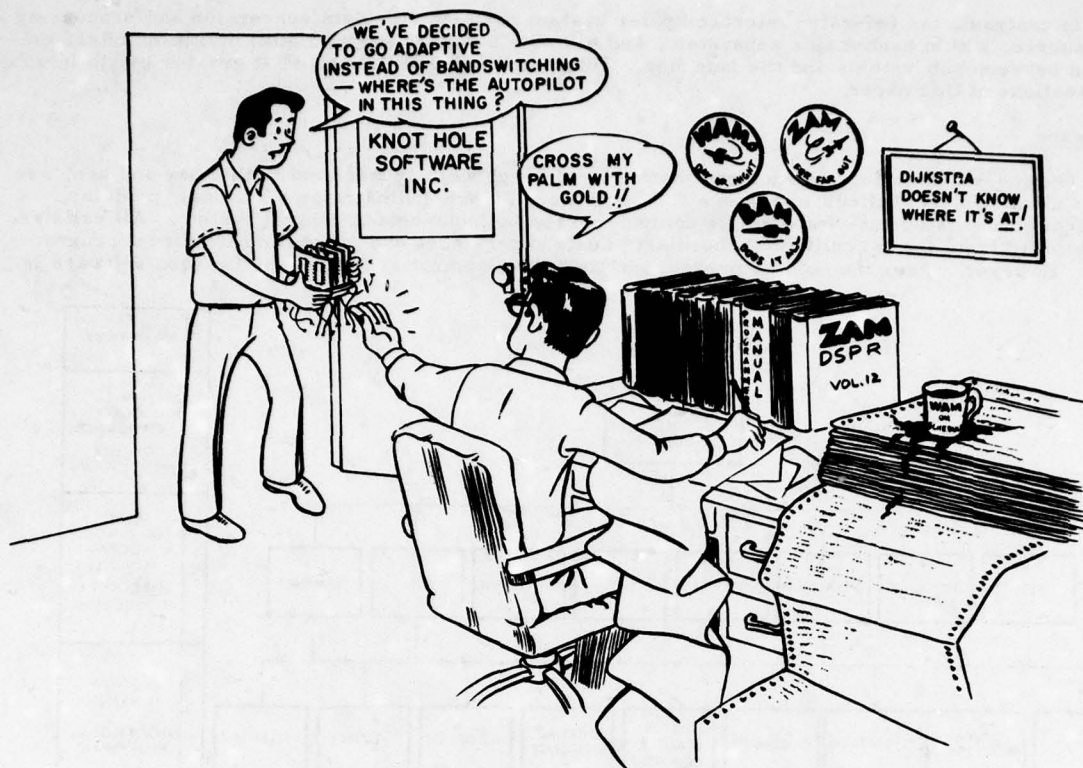


Figure 3 - Single Computer System Software is Inaccessible to Subsystem Designers

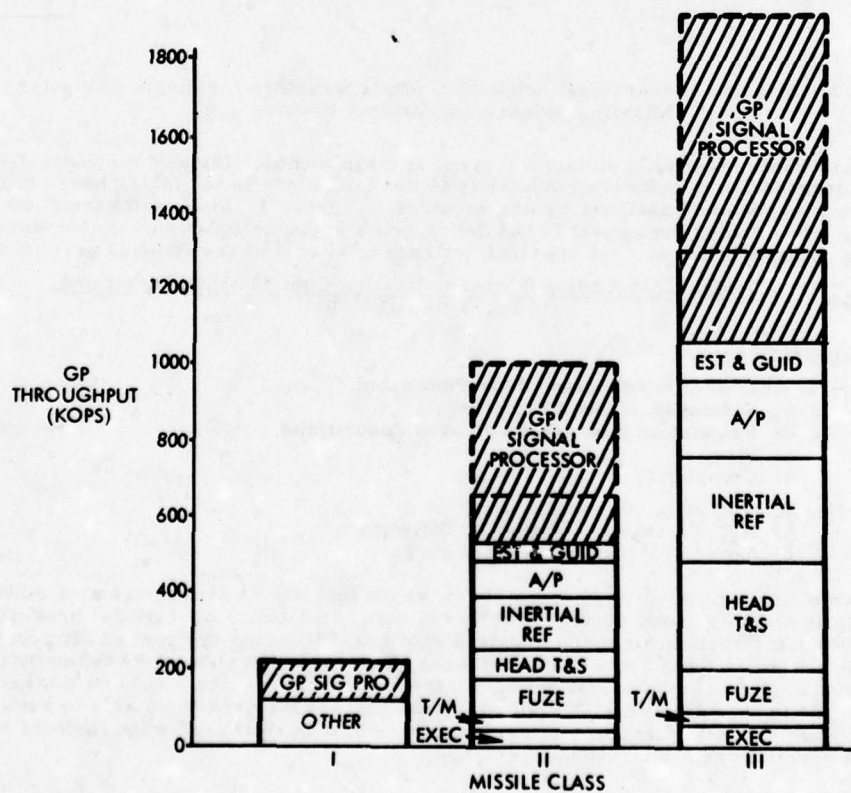


Figure 4 - Single Computer System General-Purpose Computer Throughput Requirements

system development cycle and later throughout the life span of the missile. Since there is a finite limit to the speed of the original computer, the increased load must either be accommodated by redesigning the machine or outriggering satellite processors to absorb the overflow, which in turn tends toward a distributed system of haphazard design.

DEFINING/IDENTIFYING SYSTEM STRUCTURES

Before embarking upon the design of any federated system it is important to consider the whole system as opposed to applying federated techniques on a piecemeal basis. The reason for this is to identify the characteristic structure of the system in terms of its constituent functions, data flow and data rates. Figure 5 shows the major functions of a typical missile system and their relationship to one another. In the system shown, the target sensor is mounted on a gimballed platform stabilized against missile body motion by platform-mounted rate gyros and torquers in conjunction with the seeker head control electronics.

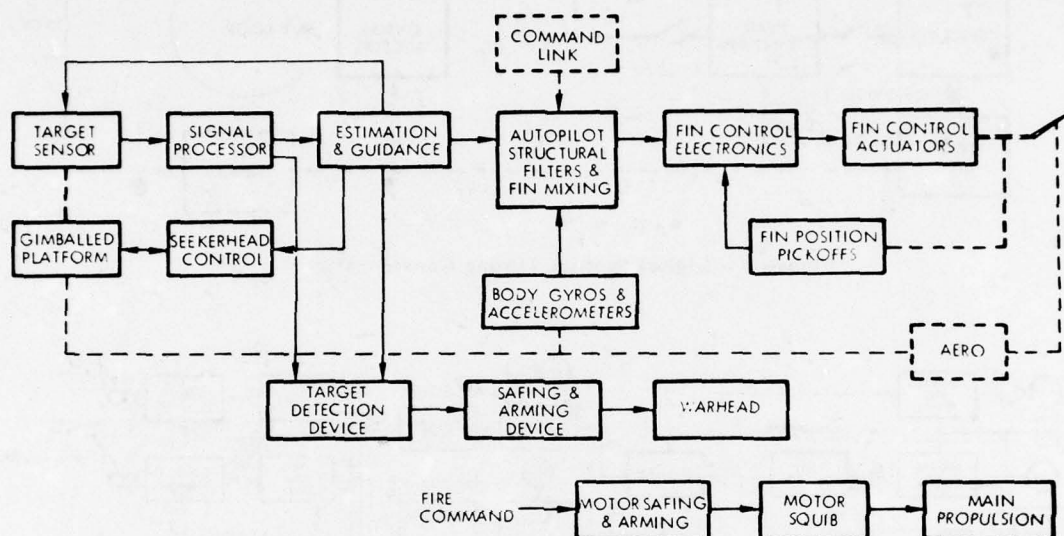


Figure 5 - Missile Guidance and Control System, Functional Block Diagram

Target sensor, e.g. radar, infrared (IR) electro-optical (EO) outputs are processed by the signal processor which provides target range and angle data, (angle only for IR and EO sensors), for subsequent filtering and processing into boresight error and 'g' commands using appropriate estimation and guidance law algorithms. The latter "steering" data controls the seeker platform and autopilot for target intercept. The autopilot also stabilizes the airframe against body motion and bending effects using body gyros and accelerometers as data sources and outputting fin deflection commands to fin control actuators. Detonation of the warhead is determined by the detection of the target by the warhead's target detection device augmented with end game geometry data from the primary target sensor signal processor and estimator. The form of motor control can vary from a simple squibbing signal from the weapon control system, via the umbilical, to sophisticated fuel control based on temperature, pressure and aerodynamic data in the case of ram-jet propulsion units.

The degree of interaction between the functional components of the system, their physical relationship, system modularity requirements, and the magnitude of the processing task in each case, influences the structure of the practical distributed microcomputer system.

System Timing Considerations

The basic or characteristic structure of a system as far as its implementation with distributed microcomputers is concerned, is determined by the system timing constraints and the autonomy of functions. Figure 6 shows the system of the previous figure with switches interposed between the major functional blocks and the associated sampling or update rates indicated to satisfy the Nyquist criteria. Three major control loops are visible, viz: seeker head, autopilot and steering command. The first two of the latter require relatively high sampling rates (125-500 Hz), to meet the bandwidths involved, whereas the steering command update rate is quite low (10-20 Hz). Also, the two high speed loops are virtually autonomous with their respective sensors and torquers/actuators.

System Parallelism

Figure 6 views the system as a set of functional blocks, but, if the system is redrawn to reflect the planar control channels of pitch and yaw for the seeker gimballed platform, pitch, roll and yaw for the autopilot, branching out into four fin control channels, then parallelism becomes evident, (Figure 7). The latter system characteristic offers potential for using several low throughput microcomputers as opposed to a few high throughput machines.

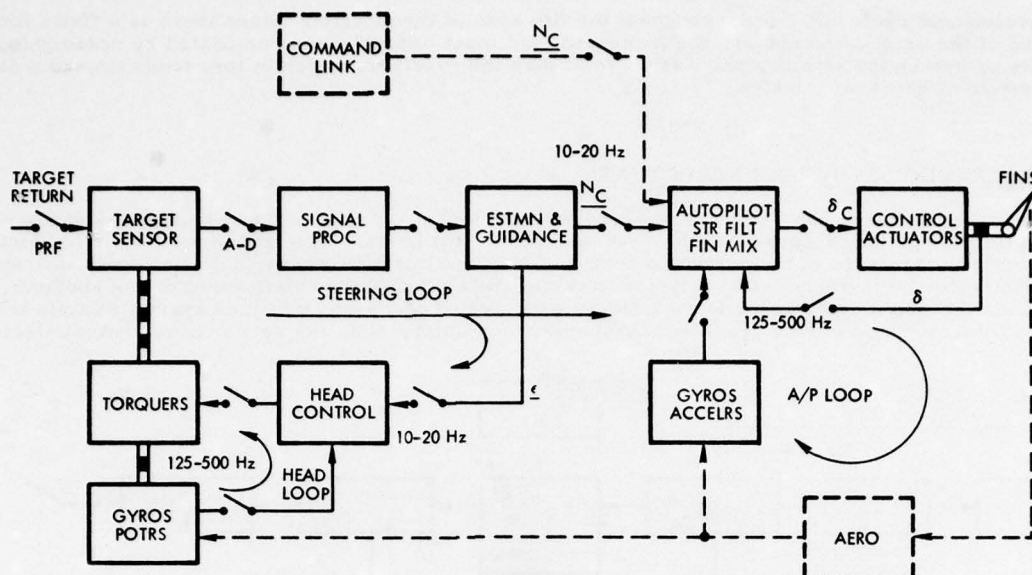


Figure 6 - Digital System Timing Considerations

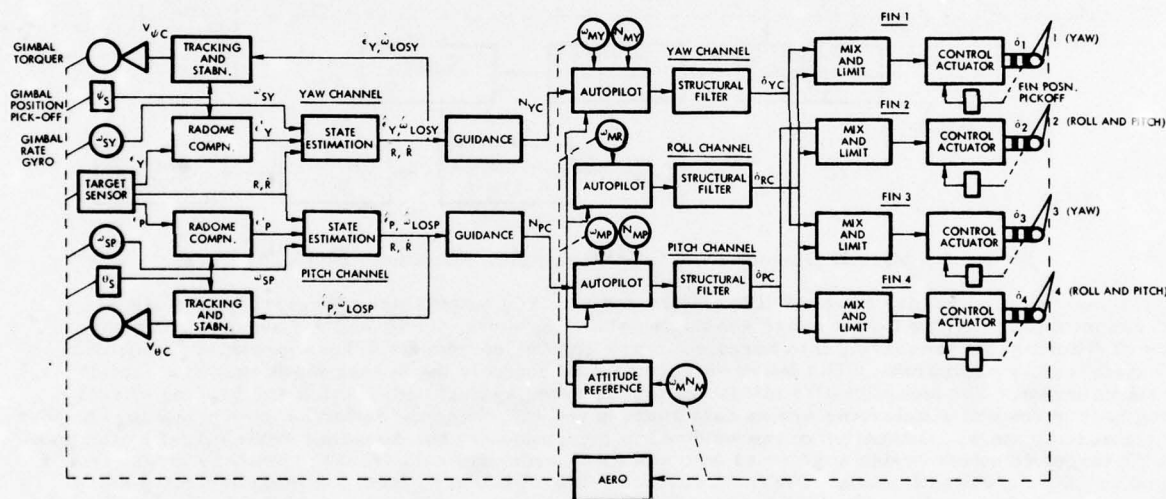


Figure 7 - System Partitioning by Control Channel

Macro-Structure System

Based upon the system as it appears in Figure 6, the obvious macro-structure which exploits subsystem autonomy and low intersubsystem data rates is as shown in Figure 8. One microcomputer is assigned to each major subsystem and a common input-output (I/O) interface interconnects subsystems via a system bus at the low 10 Hz update rate. In terms of control hierarchy, the target seeker microcomputer controls the system bus since all other subsystems are subordinate "users" of the seeker data, (Figure 9). This form of distributed microcomputer system is a true federated system, since each microcomputer operates virtually autonomously. Further, it meets the subsystem modularity design goal whether subsystems are colocated physically or not. However, there is one major drawback to this level of partitioning, as shown in Figure 10. Throughput requirements for the individual microcomputers vary widely from up to one Mops to as low as 50 Kops. As a result, the high throughput requirements of the seeker, flight control and head control functions indicate a bit-slice Schottky-bipolar or complementary metal-oxide semiconductor, silicon-on-sapphire (CMOS-SOS) device technology machine, and the remaining low throughput functions a single-chip microcomputer. The cost of designing and building the bit-slice μ Cs and necessary support software is something to be avoided if possible, and hence the need arises to explore alternative implementations using one type of microcomputer-on-a-chip throughout the system.

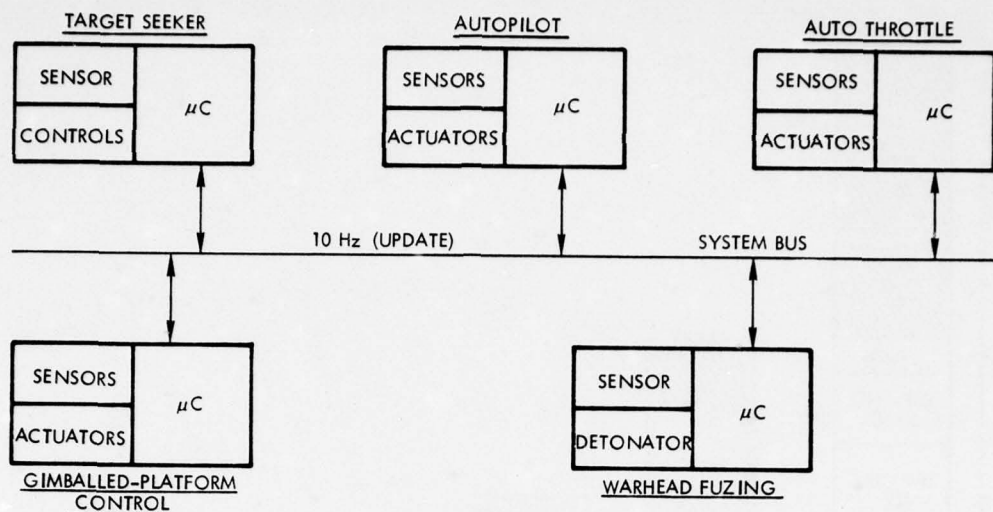


Figure 8 - Macro-Structure Partitioning

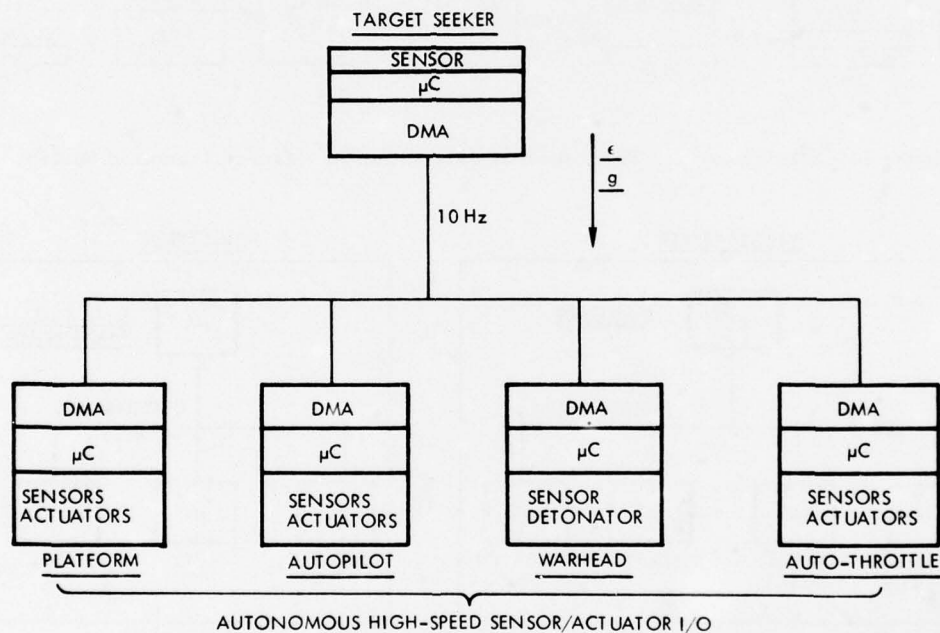


Figure 9 - Macro-Structure Control Hierarchy

Super Federated Systems

The high throughput macro functions identified in the previous paragraphs have the potential of being broken down into "microstructures"¹⁷ exploiting the intrinsic parallelism and overlap timing characteristics of the system. Figure 11 illustrates the use of separate single-chip microcomputers for each subfunction within the major functions of target seeker and autopilot.

In the case of the target seeker processing group a "heel-to-toe" computing sequence is evident since each microcomputer is waiting for the output of a preceding subfunction. However, certain preliminary operations can proceed while waiting for real-time update information e.g. state estimation. Further, since the spectrum analysis subfunction is a fixed entity i.e., either a 64, 128 or 256-point fast Fourier transform (FFT) process, then this should be executed in a high-speed special-purpose processor to allow more time in the overall budget of 20 or so milliseconds for the slower general-purpose μ Cs to execute their respective tasks. In other words, software is used where flexibility is required.

The autopilot case is quite different. As was noted earlier, (Figure 7), three-axis control can be exploited through parallel processing, thereby allowing several relatively low speed μ Cs to be used to perform a high-speed composite function.

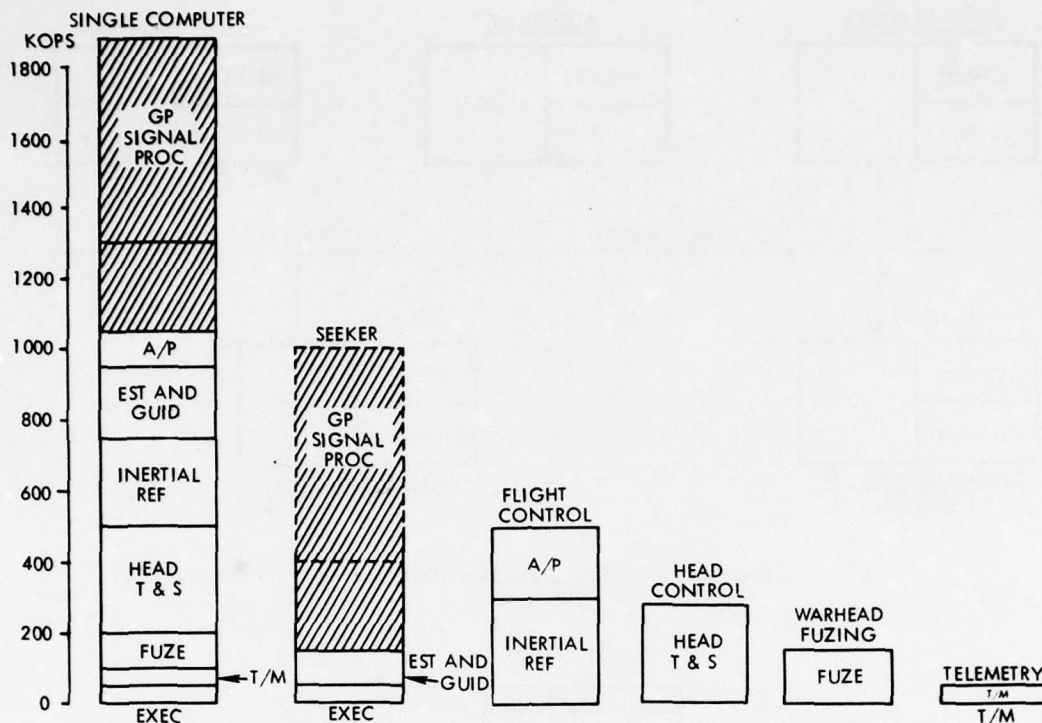


Figure 10 - Microcomputer Throughput Requirements for Macro-Structured System

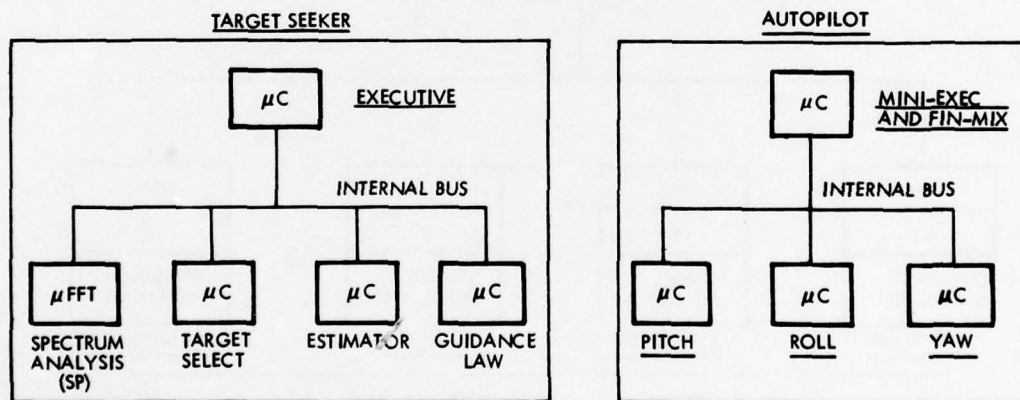


Figure 11 - Super-Federated Subsystem Processing, Target Seeker and Autopilot

Software modularity is enhanced in each of the above cases, since the functional program modules shown in Figure 2 are now visible as separate single-chip microcomputers. Taken to an extreme a 1:1 correlation between the program modules of Figure 2 and μ Cs would ensure software modularity and provide a fixed hardware interface between software routines. Subroutine calls would then be handled by hardware linkages between μ Cs. The situation depicted in Figure 3 could conceivably be transformed into the more desirable state of affairs shown in Figure 12, where a subfunction change is performed by the simple replacement of a single-chip microcomputer with the correctly programmed alternative.

MICROCOMPUTER MODULARITY

Early in the ONR study a set of microcomputer macromodules was defined^{3-7, 10} to cover the range of missile throughput requirements for a macrolevel of federation (Figure 10). Figure 13 illustrates the set of macromodules identified and Table 1 gives a brief description of them. Figure 14 shows the grouping together of modules to form a federated missile guidance and control system. The crux of modularity at the microcomputer level was the definition of a standard microbus⁴ oriented toward standard-industry semiconductor memory circuit interfaces, i. e., read-write/random-access memories (RAMs) for data storage and read-only memories (ROMs) for programs.



Figure 12 - Software Change by Hardware Substitution

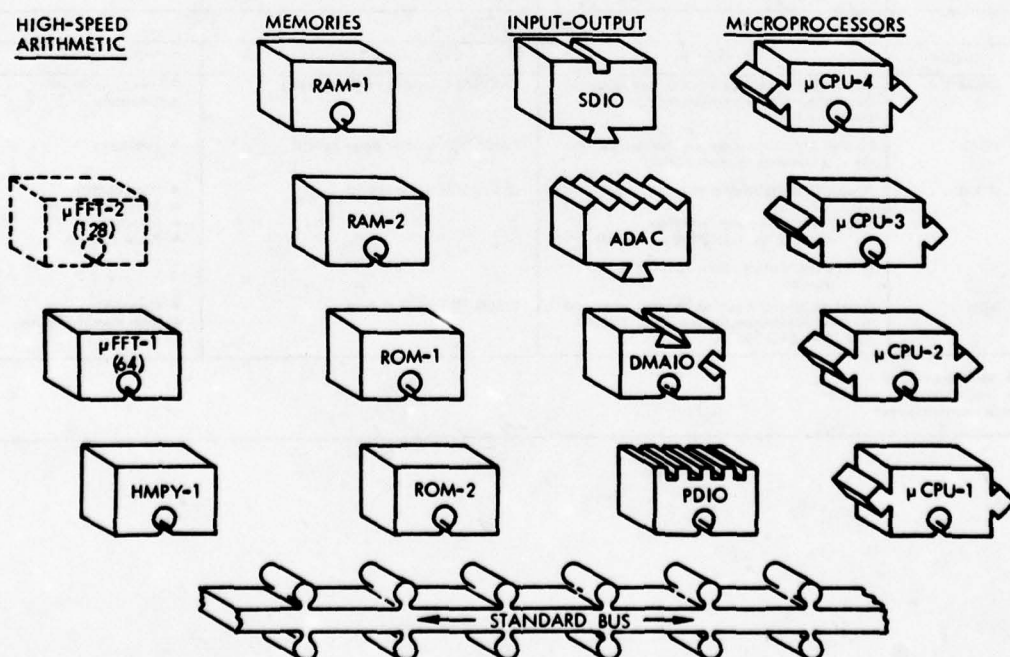


Figure 13 - Family of Microcomputer Macro Modules

TABLE 1
MICROCOMPUTER MACROMODULES

MICROPROCESSORS			
Module	Description	VLSI Circuit Technology	Application
1. μ CPU-1	Microprocessor central processing unit, 8-bit-byte general register; 2 μ s R-R, Add	N-MOS, CPU-on-a-chip, (MIL-8080)	<ul style="list-style-type: none"> Telemetry Fuzing Head control Autopilot
2. μ CPU-2	Microprocessor CPU, 8-bit byte, general register, 600 ns (8080 Emulator)	CMOS-SOS, bit-slice RALU and μ CPU hybrids (2900/3000 series or equivalent)	<ul style="list-style-type: none"> Autopilot Head control Fuzing
3. μ CPU-3	Microprocessor CPU, 16-bit word, fixed point, general register; 600 ns R-R, Add	CMOS-SOS, bit-slice RALU and μ CPU hybrids (2900/3000 series or equivalent)	<ul style="list-style-type: none"> Autopilot (Adaptive)
4. μ CPU-4	Microprocessor CPU, 16-bit word, fixed and floating point, general register; 600 ns R-R, Add (1.0 to 3.25 μ s flt. pt.)	CMOS-SOS, bit-slice RALU and μ CPU hybrids (2900/3000 series or equivalent)	<ul style="list-style-type: none"> Signal processing Estimation Guidance
HIGH-SPEED ARITHMETIC AND MEMORIES			
Module	Description	VLSI Circuit Technology	Application
5. HMPY-1	Hardware multiplier, 200 ns, 16 x 16-bit multiply	CMOS-SOS single hybrid	<ul style="list-style-type: none"> Throughput enhancement for μCPU, e.g., Class I signal processing
6. μ FFT-1	Micro Fast-Fourier Transform processor, 40-400 μ s for 64 points, 8 + J8	CMOS-SOS or CCD RALU and μ PCU hybrids (2900 series or equivalent)	<ul style="list-style-type: none"> Throughput enhancement for μCPUs, e.g., Class II and III signal processing
7. RAM-1	Random-access, read/write memory, medium speed, 128-2 K bytes, 500 ns max. access time	N-MOS DIP hybrid	Data <ul style="list-style-type: none"> Telemetry Fuzing Head Control Autopilot
8. P/PROM-1	Programmable (mask/electrically) read-only memory, medium speed, 1K-16K bytes, 500 ns max. access time	N-MOS DIP hybrid	Programs <ul style="list-style-type: none"> Signal Processing Estimation Head Control Autopilot Fuzing
9. RAM-2	Random-access, read/write memory, high speed, 256-1K x 16 bits or 256-2K bytes, 100 ns max. access time	CMOS-SOS DIP hybrid	Data <ul style="list-style-type: none"> Signal Processing Estimation Head Control Autopilot Fuzing
10. P/ROM-2	Programmable (mask/electrically) read-only memory, high speed, 1K - 4K x 16 bits or 1K-8K bytes, 100 ns max. access time	CMOS-SOS DIP hybrid	Programs <ul style="list-style-type: none"> Signal Processing Estimation Head Control Autopilot Fuzing
INPUT - OUTPUT			
Module	Description	VLSI Circuit Technology	Application
11. DMAIO	Direct-memory-access input-output channel, parallel word/byte transfers to/from micro-computer RAM	CMOS-SOS bipolar single hybrid	All microprocessor applications
12. PDIO	Parallel digital input-output channel, parallel discrete transfers to/from μ CPU	CMOS-SOS bipolar single hybrid	<ul style="list-style-type: none"> Telemetry
13. ADAC	Analog-to-digital/digital-to-analog input-output channel A-D: 8/16/24 chs. sim. S/H mux 8/10/12-bit. A/D 3/6/8 μ s max/ch D-A: 8 chs. demux., S/H, 12-bit D-A, 5 μ s max/ch.	CMOS-SOS single hybrid	<ul style="list-style-type: none"> Head control Autopilot Telemetry Radar receiver
14. SDIO	Serial digital-input-output channel, word and bit serial-data/command transfers. 1Mbit/s max., MIL-STD-1553A	CMOS-SOS single hybrid	<ul style="list-style-type: none"> Avionics Inter microcomputer
S/H - sample-and-hold Mux - multiplexer Demux - demultiplexer			

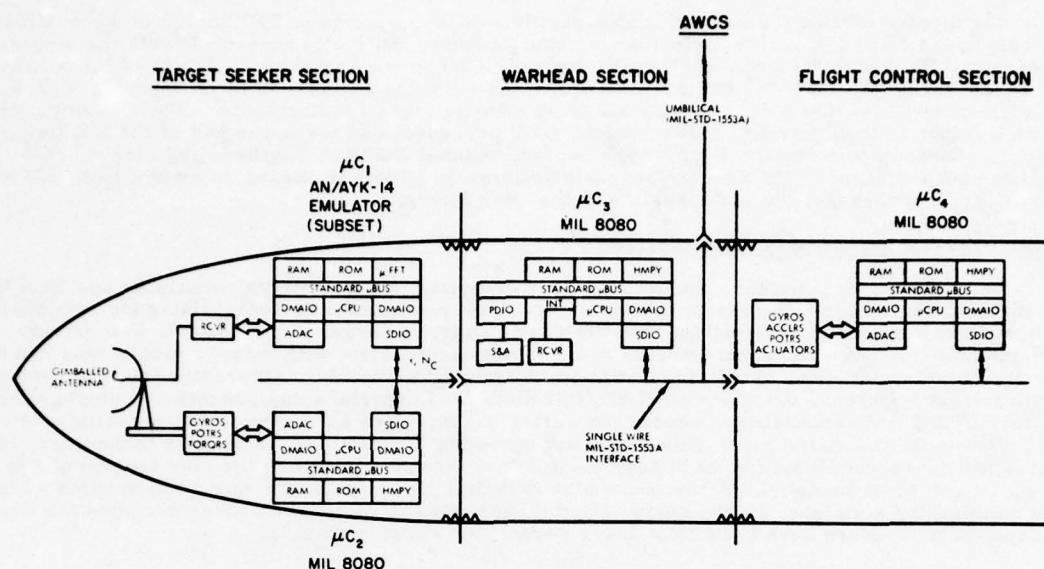


Figure 14 - Modular Federated Microcomputer Missile Guidance and Control System

Programmable Microbus Interface Module

Through the definition of a programmable microbus interface module (MIM),¹² using high-speed field programmable logic arrays (FPLAs) and programmable read-only memories (PROMs), standard-industry microcomputer components, i. e., microprocessors, RAMs, ROMs, multiplexer A-D converters, D-A converters and serial digital I/O modules, have been readily integrated into a desired microcomputer configuration. Further, each individual component can be replaced with a more desirable product from a different manufacturer, at any time during the life cycle of the system, by reprogramming the MIM to accommodate the interface peculiarities of the new product. Whereas there are several standard microbuses in the industry today, e. g. Intel MULTIBUS, S-100 Bus, National Microbus, European MUBUS, etc., no two busses are exactly the same. The latter situation gives rise to a wide variety of component interfaces and tends to restrict the adoption of new products and their integration into an existing microcomputer without significant interface redesign.

Spectrum Analyzer Module

Missile radar target seeker signal processing requirements are low compared to avionic and ground-based air defense systems (Figure 15). However, the μ FFT module of Figure 13 using bit-slice microprocessor circuits requires approximately 150 LSI/MSI/SSI circuits, dissipates approximately 50W,

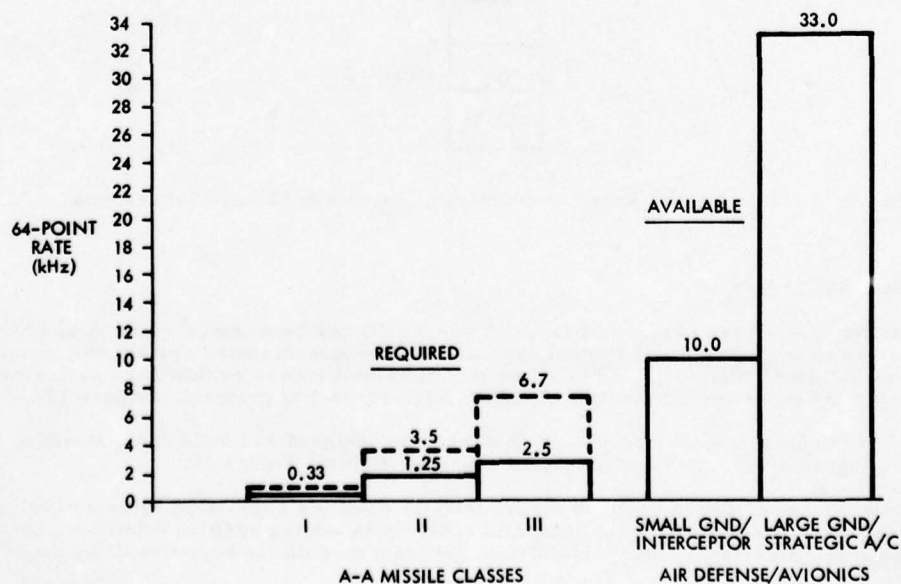


Figure 15 - Radar Signal Processing Throughput Requirements

using Schottky bipolar circuit technology, and executes a 64-point complex FFT in 300 μ s approximately, meeting only Class I and II missile performance requirements. Such a processor dwarfs the single-chip microcomputer. In contrast, a charge-coupled device (CCD) processor using the chirp-Z transform (CZT) and transversal filters¹³⁻¹⁵ executes an equivalent 64-point analysis in 13 μ s approx., with a power dissipation of less than 5 W, meeting all three missile class requirements. While dark current is a limiting factor in the dynamic range of analog CZT processors at the upper end of the MIL temperature range, recent improvements in prototype surface channel CCDs at Raytheon and elsewhere¹⁶ indicate a temporary situation in this performance deficiency. Further, based on recent NASA/TI work, a 2-chip CCD CZT processor appears feasible in the near future.

Serial-Digital Input-Output (SDIO) Module

The SDIO module provides a MIL-STD-1553A-compatible serial digital multiplex bus interface between microcomputers in the missile and the external weapon control system. Using conventional transformer coupling to the transmission line requires relatively large, high-current, line driver, receiver and transformer components which, in turn, are inconsistent with today's single-chip microcomputers and the small size, weight and power limitations of a missile. Fiberoptic coupling between subsystem microcomputers, using simple LED/PIN diode/T²L interface components and single-chip Manchester II/NRZ code convertors, reduce the serial I/O interface hardware to more realistic proportions. However, the single party-line bus is not currently amenable to fiber-optic technology, since T-couplers introduce a 3 dB loss at each drop point. A simple alternative is the ring system of Figure 16, using a round-robin protocol. A more complex multiline approach is the star configuration which would be suitable for a simple, single-mode, short-range missile where the seeker becomes the focal-point. Eight-port couplers have been built under recent Air Force contracts.

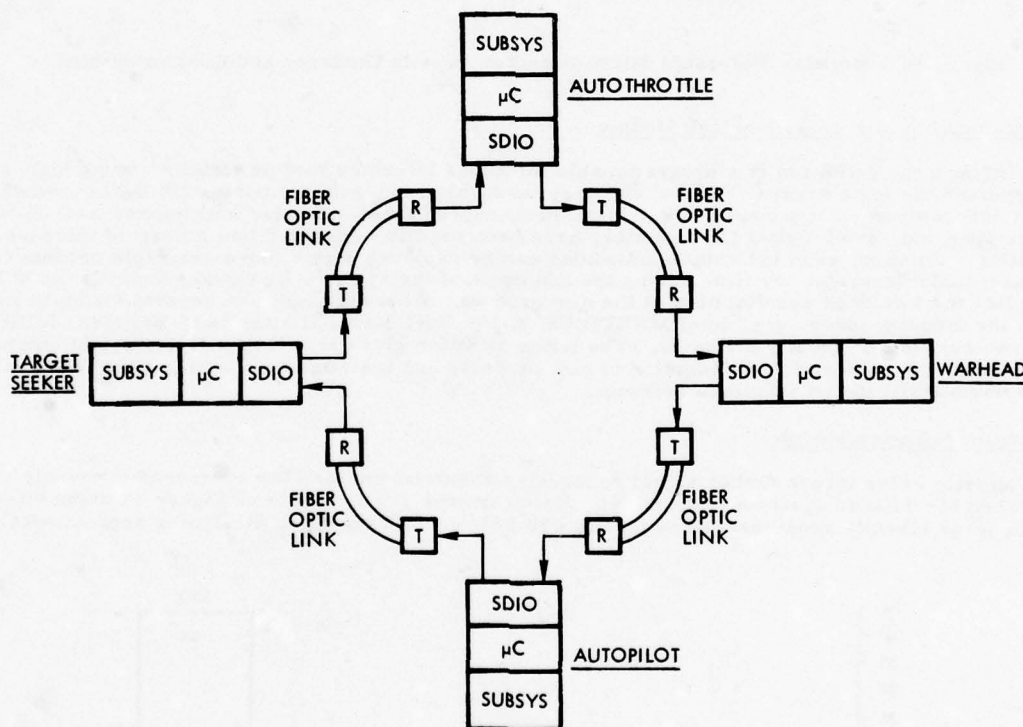


Figure 16 - Fiber-Optic Ring Communications Between Missile Subsystems

DEMONSTRATION SYSTEM

The culmination of the work performed for ONR and NSWC has been the fabrication of a basic federated microcomputer guidance and control system. This microcomputer system will constitute the "hardware-in-the-loop" element of a real-time missile simulation to evaluate the performance of the federated approach under the constraints of a MIL-STD-1553A I/O protocol, (Figure 17).

Breadboard versions of the μ C macromodules have been designed and built using standard industry μ C components integrated with microbus interface modules (MIMs), Figure 18.

The simulation is based upon a modular digital missile guidance simulation system developed for NSWC under a separate contract. System growth is achieved by adding additional microcomputers to the system bus and transferring/recoding simulation program modules to be executed by the appropriate microcomputer(s).

Figure 19 shows modular growth from a simple guidance and control system to a super-federated system using several microcomputers of the same type and maintaining the original memory and I/O modules.

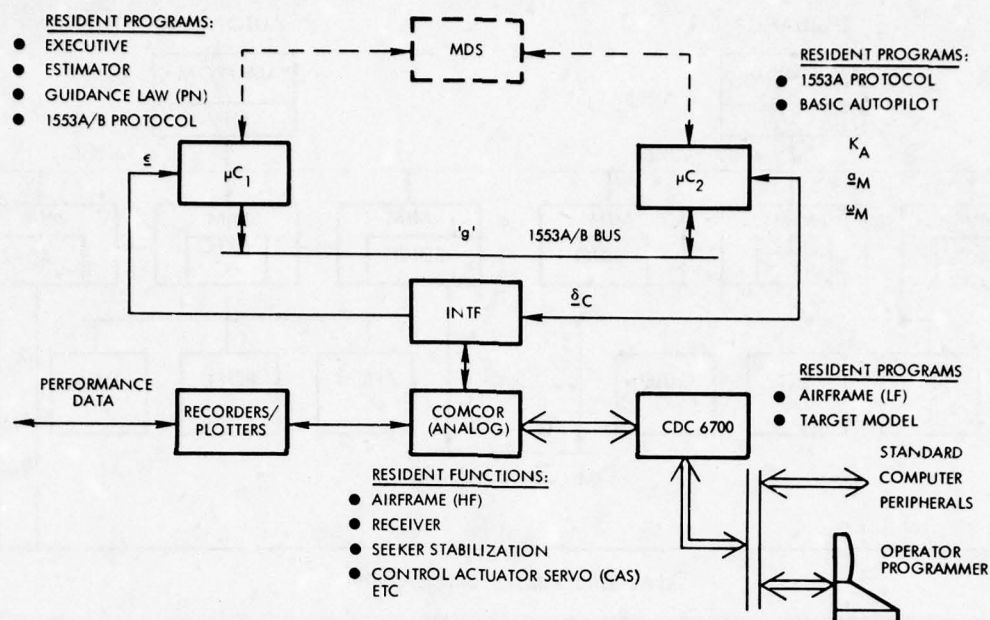


Figure 17 - Basic Hardware-in-the-Loop Federated μC System for Missile Performance Simulation

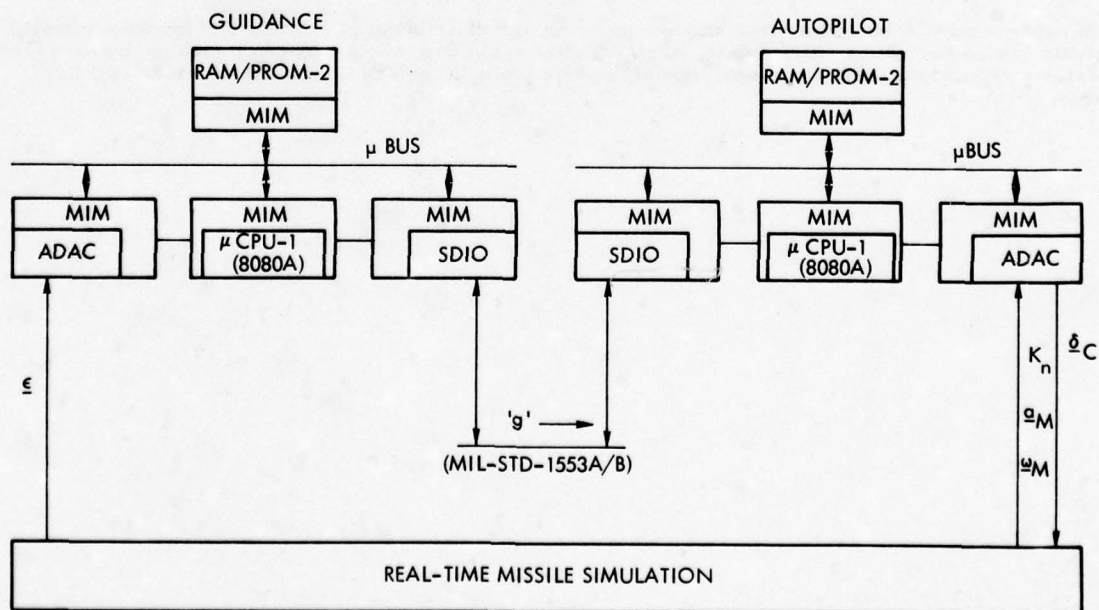


Figure 18 - Federated Microcomputer Macromodules Using Programmable MIM Interfaces

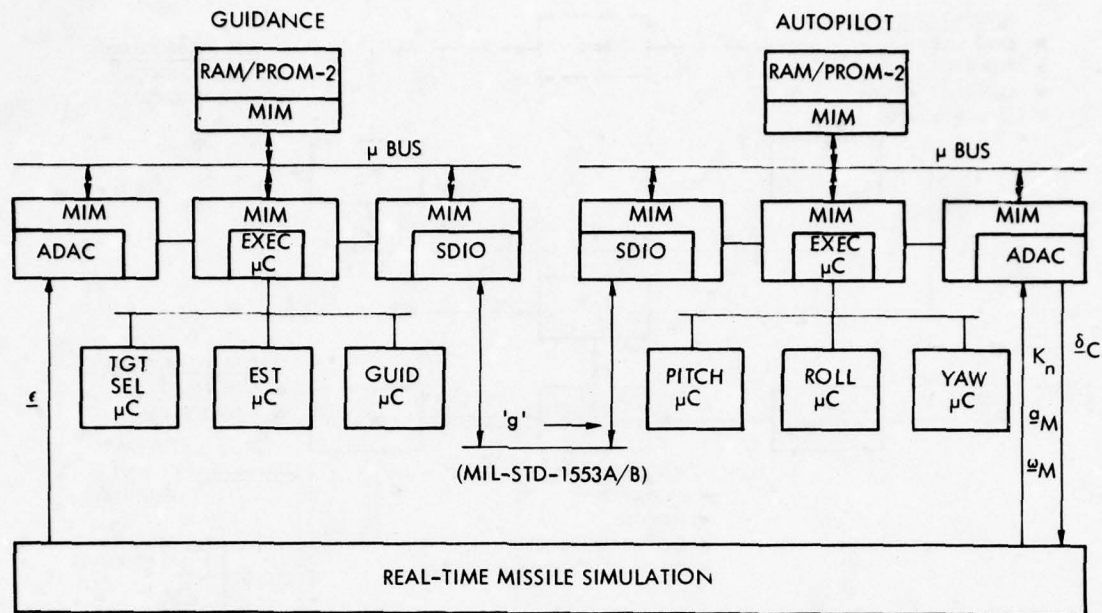


Figure 19 - Super-Federated Microcomputer System for Higher Performance Missile Guidance and Control

CONCLUSIONS

Federated microcomputer systems provide the flexibility to design, develop, modify and update missile guidance and control systems on an individual subsystem basis, thereby enhancing system modularity. Standard-industry microcomputer components which meet military environmental specifications can be integrated into a set of microcomputer macromodules using a standard programmable interface module and microbus. To achieve and maintain modularity in software, the potential exists to assign each major program module to a separate single-chip microcomputer, placing a fixed hardware interface between major function algorithms. Furthermore, by exploiting parallelism and/or the time overlapping of function execution, the use of several standard-industry, single-chip microcomputers in a "super-federated" configuration eliminates the need to resort to one-of-a-kind, high-speed, bit-slice processors, for high-performance missiles, with their attendant hardware and software logistics support problems. In terms of signal processing, improved charge-coupled device technology, in the form of chirp-Z transform processors using transversal filters, offers a solution to the high chip/parts count of current fast Fourier transform processors. A two-chip CZT processor would match the level of large-scale circuit integration presently available in microcomputer technology.

In cases where federated microcomputer systems are distributed physically throughout a missile, relatively low performance, fiber-optic, serial-digital communications between microcomputer-based subsystems eliminates the high-power, transformer-coupled interface of traditional electrical bus systems.

REFERENCES

- 1) Hall, B. A., and Trainor, M. V., "Modular Digital Missile Guidance System Study," Ph. I Report, 30 June 1974, DDC-AD-784-969/8GA.
- 2) Langley, F. J., Cooney, J. J., "Synchronous Microcomputer System for On-Board Missile Guidance and Control," 1975 National Computer Conference, Anaheim, CA, 19-22 May 1975.
- 3) Hall, B. A., and Langley, F. J., "Modular Digital Missile Guidance," Ph. II Report, 28 January 1976, DDC-AD-B010399L.
- 4) Langley, F. J., "Macro-Modular Microcomputer Family for On-Board Missile Guidance and Control," IEEE/NADC Warminster, 22-24 June 1976.
- 5) Hall, B. A., Langley, F. J., and Wefald, K. O., "Computer Design Requirements for Digital Air-to-Air Missiles," AIAA Guidance and Control Conf., San Diego, CA., 16-18 August 1976.
- 6) Langley, F. J., "Modular Digital Missile Guidance System Study," Ph. III Report, 4 May 1977, DDC-AD-A042466.
- 7) Langley, F. J., "Macro Modular Microcomputer Family for Digital Missile Guidance and Control," DDR&E/IDA Symposium on the Utilization of LSICs in Military Systems, Arlington, VA, 9 August 1977.
- 8) Nesline, F. W., and Langley, F. J., "Computer Architecture for Digital Control of Homing Missiles," SAE Aerospace Control and Guidance Systems Committee, Ft. Walton Beach, October 1977.
- 9) Langley, F. J., "The Application of Microcomputers to Digital Missile Guidance and Control," 1977 Mini/Micro Computer Conference and Exposition, Anaheim, CA, December 1977.
- 10) Langley, F. J., Sheehan, J. and LaGro, G. A., "Simulating Modular Microcomputers," 11th Annual Simulation Symposium, Tampa, FL, March, 1978.
- 11) Langley, F. J., and Cruickshanks, A., "Microcomputer-Based, On-Board Missile Guidance and Control," IEEE Guidance and Control Group, Boston Section, MITRE, Bedford, MA, May 1978.
- 12) Langley, F. J., and Demetrick, J., "An Adaptive Microprocessor/Microbus Interface Module," IEEE Annual Microprocessor Workshop, John Hopkins University, APL, Laurel, MD, June 1978.
- 13) Langley, F. J., Goldstein, D. S. and Mannion, A. J., "Real Time Signal Processing Devices for Missile Guidance and Control," 22nd SPIE Symposium, San Diego, CA, August 1978.
- 14) Zimmerman, T. A., and Barbe, D. F., "A New Role for Charge Coupled Devices: Digital Signal Processing," Electronics, 31 March 1977.
- 15) Rabiner, L. R., Schafer, R. W., and Rader, C. M., "The Chirp Z-Transform Algorithm," IEEE Trans. on Audio and Electroacoustics, Vol. AU-17, pp. 86-92, June 1969.
- 16) Claeys, C. L., et al, "Elimination of Stacking Faults for Charge-Coupled Device Processing," Proceedings of the Third International Symposium on Silicon Material Science and Technology, Electrochemical Society, May 1977.
- 17) Langley, F. J., "Applications of Microprocessing in Distributed Systems," AIAA/DPMA Microprocessor/Microcomputer Applications Conference, Newport Beach, CA, October 1978, Washington, D. C., 3 April 1979 and Boston, MA, 1 May 1979.

COPRA

UNE LIGNE NOUVELLE DE CALCULATEURS RECONFIGURABLES ULTRA-FIABLES DESTINEE AUX APPLICATIONS AEROSPATIALES EMBARQUEES

C. MERAUD ET F. BROWAEYS
SAGEM - 6, AVENUE D'ILENA, 75783 PARIS CEDEX 16 - FRANCE

RESUME

COPRA (Calculateur à Organisation Parallèle Reconfigurable Automatiquement) couvre une gamme allant du mono-processeur redondant (200 kops/sec) aux multiprocesseurs reconfigurables (400, 600 et 800 kops/sec).

Une maquette a déjà été réalisée et un prototype est en développement. Celui-ci correspond à une configuration double-duplex capable de survivre au moins à toute première panne ainsi qu'aux pannes transitoires de manière transparente à la programmation, ce qui simplifie considérablement le développement du logiciel d'application.

Les dispositifs microprogrammés de détection de faute, de reprise automatique et de reconfiguration, sont exposés. Quelques aspects technologiques, notamment l'emploi de CMOS sur SOS, sont discutés.

Le logiciel est structuré et rigoureusement cloisonné dès le niveau machine. Des outils sont prévus au niveau superviseur pour faciliter la programmation d'une dégradation douce de la mission.

Une modélisation markovienne permet une estimation précise de la fiabilité qui, pour la configuration en développement, sera voisine de 95% sur 5 ans avec un taux d'erreurs indétectées de l'ordre de 10^{-10} par heure.

INTRODUCTION

Les unités centrales conventionnelles permettent de réaliser des systèmes de contrôle de processus de plus en plus complexes tant que le niveau de fiabilité demandé n'est pas trop élevé. Dans les applications sans configuration dangereuse (centraux téléphoniques par exemple), la disponibilité peut être améliorée en doublant l'unité centrale et en maintenant si possible l'ensemble sous surveillance humaine. Au-delà, il faut faire appel aux techniques FTC (Fault Tolerant Computer). C'est le cas désormais pour les systèmes de conduite intégrés envisagés pour les avions nouveaux (contraintes de sécurité supérieure à 10^{-10} par heure) et pour les calculateurs embarqués sur satellites pour lesquelles on exige une disponibilité supérieure à 95% sur plusieurs années.

La ligne de calculateurs COPRA en développement est destinée à ces applications. Elle exploite les techniques FTC les plus récentes. Celles-ci concernent l'implantation de dispositifs de détection des erreurs et pannes ; le recouvrement de ces dernières par reconfiguration ; la reprise automatique des traitements interrompus ; enfin la génération de logiciels sûrs.

Pour assurer la fiabilité spécifiée sur les tâches les plus critiques, il faut en effet :

- insérer des redondances pour assurer la survie des fonctions du calculateur et garantir l'intégrité de l'information mémorisée,
- assurer la sécurité des calculs (sans complexifier les logiciels), pour éviter la production d'événements catastrophiques.

L'unité centrale COPRA satisfait ces contraintes et permet la programmation sûre de logiciels importants. Si une faute apparaît, sa détection et son confinement sont réalisés avant qu'une pollution irréversible de l'information n'ait lieu qui empêcherait une reprise correcte des opérations. Ces dernières opérations sont donc réalisées avec un matériel distinct pour que ses défaillances n'altèrent pas le traitement en cours.

Le cas de pannes transitoires étant le plus fréquent (99% des défaillances programmes), celles-ci sont filtrées au niveau matériel par une opération de reprise automatique de la séquence en cours.

Si la panne est confirmée, le module siège de la cause origine est isolé, ce qui active le module de secours. On exécute ensuite une reprise automatique du traitement.

Ces opérations sont réalisées avec simplicité grâce à la technique de microprogrammation employée par la SAGEM pour la réalisation de toutes ses unités centrales depuis 1967. Cette technique permet de disposer à la fois :

- d'une liste d'instructions adaptée au domaine d'application,
- des instructions de diagnostic indispensables à l'exécution des tests périodiques nécessaires à l'élimination efficace des pannes cachées,
- des opérations logiques spécifiques au traitement des pannes et à leur recouvrement.

L'amélioration de la fiabilité du logiciel s'appuie sur un découpage en tâches et un arrangement de l'information en mémoire qui soit à l'image de la structure fonctionnelle de l'application. Les règles de

communications et d'accès entre les parties peuvent alors être décrites et permettre la vérification statique et dynamique de leur respect.

Il est ainsi possible de réaliser un cloisonnement efficace et surveillé pour chaque sous-système. Il est également possible d'accroître la finesse du découpage en tâches fonctionnelles surveillées pour exploiter au mieux les possibilités de dégradation douce de l'application.

Ces caractéristiques rétablissent une ségrégation claire des fonctions et sous-fonctions tout à fait comparable à ce qui est classiquement réalisé dans les structures analogiques redondantes à composants discrets. L'absence de cette ségrégation sur les calculateurs conventionnels explique en grande partie la réserve constatée vis-à-vis de l'informatique, même quand elle est fiabilisée contre les pannes matérielles.

En conclusion, l'utilisation du calculateur FTC COPRA permet grâce à sa très grande fiabilité de structure et à ses caractéristiques logicielles, l'écriture dans de bonnes conditions du logiciel d'application car on n'a pas à tenir compte de l'arrêt ou des fautes du calculateur. La détection des fautes et la reconfiguration automatique protègent la mission des conséquences d'un arrêt du calculateur ou de son fonctionnement anarchique et permet d'atteindre ainsi d'une façon rigoureusement évaluable le niveau de fiabilité requis pour l'ensemble de l'application.

Tous les points résumés dans cette introduction vont être développés dans ce qui suit.

HISTORIQUE

Le développement du COPRA est le résultat d'une collaboration de six années entre les sociétés SAGEM maître d'oeuvre, et Electronique Marcel Dassault. Ces deux compagnies conçoivent et réalisent des calculateurs temps réel depuis le début des années 60. Toutes les deux ont très tôt essayé d'améliorer la sûreté de fonctionnement des calculateurs par insertion de redondances à une époque où le niveau d'intégration ne dépassait pas la taille d'un registre (études SAPHIR et MECRA).

La fin des années 60 voit l'émergence des techniques LSI qui conduit au concept d'une redondance plus globale, avec une fiabilité plus élevée pour un volume et un coût raisonnables.

C'est dans ce contexte que les deux sociétés rassemblent leurs efforts sur le projet COPRA avec l'aide du Ministère de la défense par le canal de la DRET.

Cet effort a abouti en 1976 à la réalisation d'une maquette de validation des solutions de base, tandis que se poursuit aujourd'hui la réalisation d'un prototype.

SURVIE AUX PANNES

1. Survie aux pannes transitoires (figure 1)

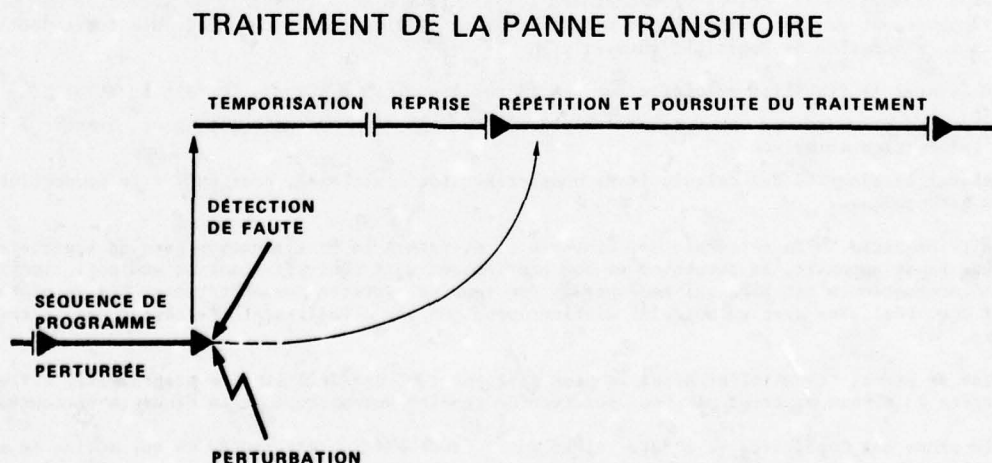


FIGURE 1

Cette survie et la différenciation de ce type de panne de celui des pannes permanentes de composants est l'un des points les plus importants dans un système qui détecte toute erreur avec une efficacité très élevée.

On admet en effet que 99% des fautes ou défaillances de programme observées sont dues à des transitoires provenant de l'environnement, ou de composant en situation de fonctionnement marginal dans certaines configurations de données.

En réalité, cette proportion serait beaucoup plus élevée sur les machines conventionnelles si toutes les erreurs frappaient des informations critiques ce qui est loin d'être le cas. Donc une machine FTC qui assure la sécurité en s'efforçant de détecter toutes les erreurs doit être capable de discerner le caractère transitoire de leur cause et survivre sans effectuer de reconfiguration prématurée.

Une solution consiste à travailler par triplication et vote ; mais cette solution en réalité complexe est vulnérable aux éblouissements et mal adaptée aux missions longues.

La seule alternative est de tirer parti du caractère séquentiel des traitements numériques pour introduire un mécanisme de reprise automatique, qui permette après détection d'une erreur sur la séquence en cours, de répéter celle-ci à partir d'un point de reprise antérieur.

Il est entendu qu'un tel mécanisme doit être transparent à la programmation, c'est à dire n'entraîner aucune contrainte pour l'utilisateur.

Pour être simple et rapide, sans exiger de matériel supplémentaire notable, le mécanisme de reprise est réalisé par microprogrammation. Il consiste, au niveau du point de reprise ("PDR"), à faire une courte sauvegarde de contexte dans la mémoire, d'une façon analogue à ce qui est fait au moment d'une interruption. L'opération inverse, effectuée après détection d'une erreur, réalise une reprise.

Dans un tel mécanisme, les points de reprise ne peuvent pas être implantés au hasard. En effet, la reprise étant effectuée avec les états auxquels la mémoire et les entrées/sorties sont rendues au moment de la détection d'une erreur, il faut que ces états soient compatibles avec une répétition identique des mêmes calculs. (Exemple : $A + B \rightarrow R, R \rightarrow A$ n'est pas répétable).

L'implantation des points de reprise est faite à la génération des programmes par un algorithme spécial. Celui-ci détermine à partir de quelle action supplémentaire, la séquence en cours d'assemblage est susceptible de modifier ses conditions initiales, et insère alors un point de reprise (Exemple : $A + B \rightarrow R$, "PDR", $R \rightarrow A$. La coupure en séquences répétables est réalisée).

Le point de reprise est un point de non retour et le départ d'une nouvelle séquence. La période de sensibilité est de quelques nano-secondes ; elle est donc quasi nulle. La distance moyenne entre points de reprise varie de 10 à 20 instructions. Ils sont préalablement armés par une sauvegarde des registres du processeur dans deux zones travaillant en bascule, la validation consistant à basculer de zone. Ces armements ont la durée d'une instruction rapide entraînant une consommation de la puissance de calcul de l'ordre de 8%.

Les reprises après détection d'erreur sont retardées d'une fraction de ms ou davantage, selon les conditions d'environnements afin d'améliorer la protection dans le cas de trains de transitoires. Leur nombre est compté, et leur fréquence comparée à des seuils à des fins de maintenance.

Un tel mécanisme libère donc bien le programmeur de tout souci de programmation pour résister à ces pannes. Ne faisant pas appel à ce dernier, aucune erreur humaine n'est à craindre dans l'insertion des points de reprise. Pour ces deux raisons, la certification du logiciel en est améliorée.

2. Détection des erreurs dans les processeurs et reconfiguration de ces derniers

Les CPU sont réalisées avec quatre microprocesseurs en tranche 4 bits du type AMD 2901 ou son équivalent ALIS en CMOS sur SOS réalisé par la société EFCIS. La largeur du mot et des chemins de données est de 16 bits. Les opérations flottantes travaillent sur des opérandes de 24 bits de mantisse et 8 bits d'exposant.

Détection des erreurs

Les unités de traitement représentant dans un passé récent, une partie importante du coût des calculateurs, SAGEM et EMD avaient étudié des solutions à base de code arithmétique pour éviter la duplication et la comparaison des UT pour la détection des erreurs.

Cette solution a finalement été écartée car insuffisamment efficace pour surveiller toutes les opérations réalisées par une unité de traitement et mal adaptées aux composants LSI actuels. Ces derniers ayant ramené aux environs de 10% la part de l'unité de traitement dans le coût d'un calculateur complet, il est devenu plus économique de revenir à la solution primitivement écartée. La solution actuelle comprend donc deux unités de traitement fonctionnant en micro-synchronisation et comparant toutes leurs sorties bit à bit à chaque micro-cycle. La comparaison elle-même n'exige que 4 composants MSI.

Reconfiguration

Au premier stade, le recouvrement de l'erreur fait l'hypothèse d'une panne transitoire. Celle-ci est recouverte par la séquence microprogrammée "Attente puis Reprise" (figure 1).

Quand la séquence persiste à rester en erreur en provenance d'UT, la panne est considérée comme permanente (figure 2). Cette confirmation provoque la coupure de l'alimentation de l'UT ce qui l'isole électriquement du système.

TRAITEMENT DE LA PANNE PERMANENTE

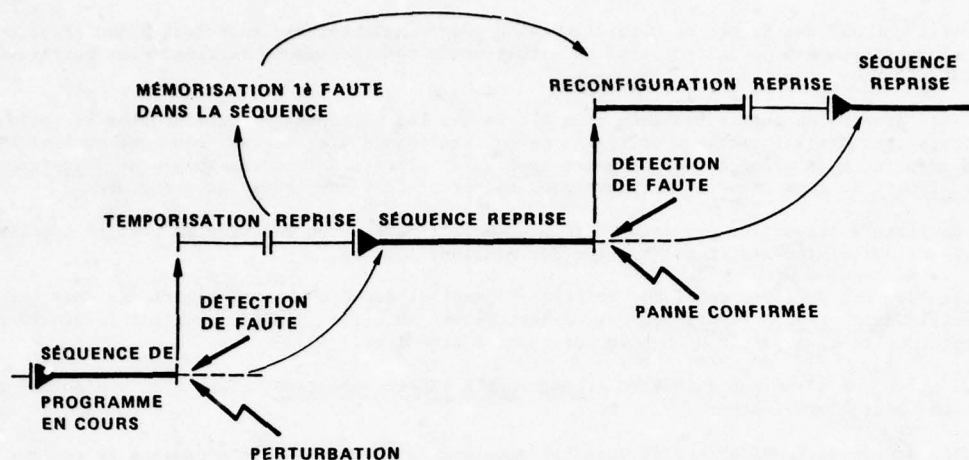


FIGURE 2

Cet état a pour effet de provoquer une interruption de niveau maximum démasquée sur une autre UT chargée de reprendre les traitements provisoirement abandonnés.

Une nouvelle distribution des tâches est effectuée par ventilation d'un nouveau jeu de masques de niveaux. L'équilibre se reconstitue progressivement à partir du fonctionnement naturel de la gestion d'un multi-processeur banalisé multiniveaux dont les niveaux sont statiquement répartis entre les processeurs.

Il faut alors que la puissance de calcul réduite laissée par la perte d'un processeur soit suffisante pour continuer à exécuter intégralement la mission. Dans le cas inverse, la suppression de tâches non critiques est envisagée pour réaliser une dégradation douce de la mission.

3. Détection des erreurs dans les mémoires et reconfiguration de celle-ci

La mémoire est réalisée avec des circuits intégrés CMOS (PROM, ROM ou RAM de capacité variée avec cependant une large majorité de boîtiers RAM 4K). Ces boîtiers sont regroupés en pages de 4K avec une page supplémentaire de réserve. L'intégrité de l'information face aux pannes est garantie par une mémorisation dupliquée dans deux blocs physiquement indépendants. L'écriture en mémoire est simultanée sur les deux blocs. La lecture a lieu avec un seul bloc à la fois, ce qui double approximativement le débit de la mémoire. Il faut au moins trois pannes parmi les boîtiers mémoires pour perdre la fonction mémoire.

Détection

En dehors des circuits de mémorisation surveillés par codage comme on va le voir, les circuits logiques de servitude de la mémoire (dialogue d'accès, décodage, adresse, reconfiguration de page) sont, pour les mêmes raisons que celles exposées précédemment, surveillés par duplication et comparaison.

Tout désaccord détecté au niveau des comparaisons, inhibe immédiatement les transferts et provoque une micro-interruption sur les processeurs pour exécuter la séquence microprogrammée "Attente puis Reprise".

Les circuits de mémorisation qui, dans le cas des grosses capacités, peuvent constituer une partie coûteuse, sont surveillés par codage. Pour cela, les circuits sont regroupés en tranches mémoire indépendantes de 1 ou 4 bits (voire 8 bits). Une tranche supplémentaire est implantée pour contenir le code détecteur. Dans cet arrangement, les pannes de composants ne peuvent provoquer d'erreur que dans une seule tranche. Ces configurations d'erreur sont parfaitement détectées tant que le système de détection n'est pas lui-même en panne. Un test périodique permet de se prémunir contre le risque de cette conjonction.

Reconfiguration

Quand une erreur est détectée, elle provoque l'"Attente puis Reprise" microprogrammée déjà citée qui filtre l'erreur s'il s'agit d'une panne transitoire.

En cas de persistance, une correction microprogrammée est réalisée à partir de la deuxième copie de l'information entretenue dans le deuxième bloc mémoire. Cette opération filtre les pannes transitoires mémorisées.

Si la faute n'a pas encore disparu, la mémoire est cette fois reconfigurée.

Cette opération s'effectue en deux temps :

- immédiatement, le processeur en dialogue avec le bloc mémoire en panne positionne une bascule d'indisponibilité du bloc qui en interdit l'usage pour tous les processeurs. Les calculs se poursuivent avec une information désormais entretenue en simple copie sur le bloc restant,

- à un instant différé, le programme périodique de test et reconfiguration de la mémoire effectue le diagnostic complet et la réparation à l'aide d'instructions microprogrammées particulières.

S'il s'agit d'une panne localisée dans une page de 4K, ce qui est le plus probable, et que la page de réserve soit encore disponible, celle-ci est substituée à la page défaillante. Ensuite, une mise à jour de l'ensemble du bloc est effectuée par recopie microprogrammée à partir de l'autre bloc suivie d'une réintroduction du bloc dans la configuration d'ensemble.

S'il s'agit d'une configuration de panne interdisant l'usage global du bloc (panne dans les servitudes logiques, collage des bus, du bloc...), ou si la page de réserve est en panne ou a déjà été consommée, la décision provisoire d'indisponibilité est définitivement confirmée par une commande de coupure de l'alimentation du bloc à partir du processeur traitant la panne.

Enfin, si le deuxième bloc mémoire est lui-même déjà en panne, ou quand la panne n'entraîne que la perte d'une page contenant des informations non critiques, on réalise une dégradation douce. Si la page perdue contient des informations critiques, ou si la panne entraîne la perte du bloc complet, on positionne vers l'extérieur l'alarme de panne complète.

A ce stade de l'exposé, il est possible de comprendre pourquoi l'utilisation d'un code correcteur ne peut pas être utilisé pour éviter de dupliquer la mémoire. La part des pannes des servitudes logiques mémoire n'est en effet pas négligeable. Il faudrait donc une redondance supplémentaire à ce niveau et qui soit précablée à la fois à l'ensemble des tranches 4 bits de la mémoire et à l'ensemble des processeurs. Cette disposition accroît considérablement la complexité au niveau des interfaces et diminue l'indépendance des causes de pannes entre les parties redondées. Pour ces raisons, l'utilisation des codes auto-correcteurs a été écartée.

4. Détection des erreurs dans les entrées/sorties et reconfiguration de ces dernières

Les entrées/sorties au niveau de l'unité centrale sont réalisées par des contrôleurs connectés sur les bus internes des blocs mémoires. Elles sont donc dupliquées. Elles se présentent pour la programmation comme des opérations de transfert dans une page particulière.

Les liaisons avec les sous-systèmes sont réalisées par une liaison série multiplexée dupliquée sur les bus de chaque bloc. Leur gestion est assurée par un canal microprogrammé existant dans chaque processeur. La fonction canal a donc les caractéristiques de détection de panne et de reconfiguration propres à la fonction processeur.

Vis à vis des pannes de l'unité centrale et jusqu'aux pannes internes des contrôleurs d'entrées/sorties on admettra que l'application des principes déjà exposés garantit la survie de la fonction entrées/sorties à de nombreuses pannes d'unité centrale et au moins à toute configuration de première panne.

Au-delà, pour survivre aux pannes des senseurs, activateurs, afficheurs, etc..., la structure COPRA fournit à l'utilisateur les moyens de connecter ces dispositifs par deux chemins indépendants, et d'être informé au niveau du logiciel de l'application de toute anomalie détectée. Il revient donc à l'utilisateur, dégagé des soucis d'une défaillance en provenance de l'unité centrale, de faire les choix judicieux en matière d'implantation de redondances dans l'ensemble de l'application et d'exploiter au mieux les informations de contexte de l'application pour obtenir la fiabilité désirée pour les entrées/sorties.

Au niveau logiciel, les outils nécessaires à une programmation efficace par l'utilisateur d'une dégradation douce de l'application ont été prévus.

5. Cas des autres fonctions

Les alimentations sont doublées, une seule d'entre elles étant en fonctionnement à un instant donné.

Les horloges, dont la fiabilité intrinsèque est déjà très élevée, sont triplées et votées à l'entrée de chaque fonction.

Il n'existe pas de point de croisement au niveau des bus interconnectant les différentes fonctions redondées qui comporte des composants ayant un taux de panne significatif.

Il a été démontré qu'aucune première panne des composants câblés sur les bus (court-circuit, collage, etc...) ou les circuits d'accès aux fonctions ne pouvaient entraîner la perte de plus d'un processeur et d'un bloc mémoire à la fois, cette dernière configuration de panne étant elle-même très peu probable.

UNE FAMILLE DE CALCULATEURS A HAUTE SURETE

En utilisant les modules de base précédemment décrits, on peut configurer les calculateurs suivants :

1. Le monoprocesseur élémentaire (UT 382-100)

Le processeur élémentaire, construit autour de quatre microprocesseurs en tranche AMD 2901 ou leur équivalent ALIS en SOS, a les caractéristiques suivantes :

- un packaging en une carte multicouche compatible avec les dimensions du 1/2 ATR court (1 MCU) ou les dimensions transverses du 3/4 ATR (6 MCU),
- débit 350 kop/s,

- 150 instructions de base et provisions pour des instructions spécifiques,
- microprogrammation sur 48 bits,
- opérations fixes sur 16 ou 32 bits. Flottant sur 24 bits de mantisse et 8 bits d'exposant.

La mémoire est réalisée avec 1 à 4 pages RAM de 1K associée à 1 à 10 pages de 2K EPROM, soit un total de 24K mots sur une seule carte au même format que le processeur. L'extension au-delà de cette capacité sur des cartes supplémentaires est possible (jusqu'à un maximum théorique de 256K mots).

Cette machine ne tolère pas les pannes.

2. Le monoprocesseur de base tolérant les pannes transitoires (UT 382-200)

A partir de la configuration de base précédente, il est ajouté :

- un deuxième processeur avec les circuits de surveillance pour constituer le processeur auto-surveillé UT 382-200 ; ce processeur à "défaillances passivées" est réalisé en deux cartes adjacentes (avec blindage électromagnétique possible),
- la mémoire précédente est complétée de ses circuits de surveillance et de stockage des bits de détection d'erreur.

Le mécanisme de reprise microprogrammée est implanté permettant à cette machine de résister aux pannes transitoires.

3. Le bi-processeur "Fault Tolerant" en développement

C'est l'organisation "double-duplex" (figure 3) typique assurant la sécurité maximum et une disponibilité très élevée.

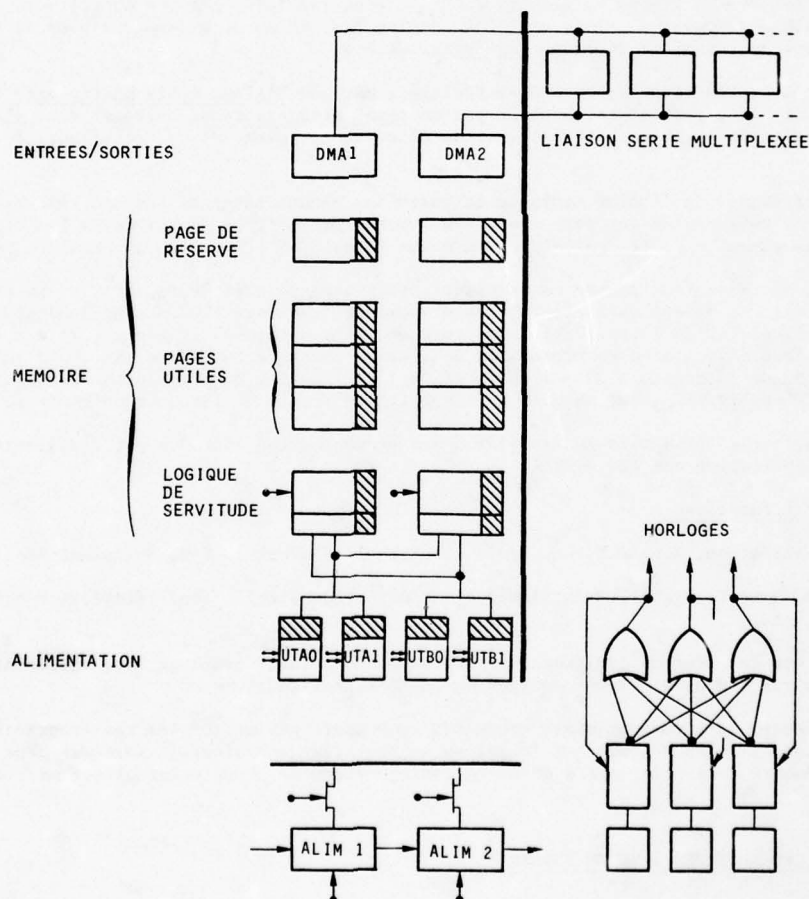


FIGURE 3

Elle comprend deux processeurs duplex, deux blocs mémoires auto-surveillés, deux contrôleurs d'entrées/sorties duplex, une horloge TMR et des servitudes doublées.

La puissance en ligne peut atteindre avec les deux processeurs allumés 600 kop/s (version T²L) ou 400 kop/s (version CMOS et SOS). En application spatiale, il est possible d'éteindre l'un des processeurs pour gagner en consommation et en fiabilité ; la vitesse est alors de 200 kop/s.

La mémoire comporte deux blocs identiques sur une carte, chacun du même type que dans l'UT 382-200.

Les entrées/sorties comprennent une liaison série dupliquée déjà décrite et des entrées/sorties discrètes.

L'ensemble est compatible avec les dimensions d'un 1/2 ATR court.

La consommation en version spatiale (CMOS sur SOS) avec un processeur allumé (200 kop/s) et 16K utiles est de 15 watts sur le 28V continu de bord.

4. Configuration multi-processeur à capacité mémoire élevée

L'organisation précédente peut être étendue à plus de deux processeurs et plus de deux cartes mémoire pour offrir des puissances de calcul élevées (800 kop/s) et/ou des capacités mémoire élevées, pourvu que les possibilités de parallélisme interne de l'application le permettent.

EVALUATION DE LA FIABILITE

Il est indispensable dans la conception d'architecture "Fault Tolerant" de prévoir une modélisation permettant de quantifier les améliorations obtenues par rapport aux machines conventionnelles non redondantes.

Ceci permet d'ajuster la configuration aux spécifications de fiabilité pour chaque application particulière.

Dans ce but, un modèle du comportement de COPRA vis à vis des pannes a été développé en utilisant les processus de MARKOV. A partir d'un état initial sans panne, l'apparition de celles-ci provoquent des transitions vers divers états possibles suivant des lois de probabilités définies et fonction du temps.

Un exemple simplifié est le suivant pour la configuration double duplex de COPRA. Celle-ci peut se trouver dans les états suivants :

- état 1 : 2 UT duplex, 2 blocs mémoires,
- état 2 : 2 UT duplex, 1 bloc mémoire (l'autre en panne),
- état 3 : 1 UT duplex, 2 blocs mémoires (1 UT en panne),
- état 4 : 1 UT duplex, 1 bloc mémoire (1 UT, 1 BM en panne),
- état 5 : détection d'une faute sur une dernière ressource ("ALARME"),
- état 6 : faute non détectée à un stade quelconque.

Si $\Pi(t)$ est le vecteur ligne $[P_1(t), \dots, P_6(t)]$ des probabilités d'avoir atteint chacun des états définis à l'instant t , l'évolution de ce vecteur satisfait l'équation différentielle suivante :

$$\frac{d \Pi(t)}{dt} = \Pi(t) \cdot \Lambda(t)$$

où $\Lambda(t)$ est une matrice carrée à 6 dimensions dont les termes sont les taux associés aux probabilités de transition d'un état à l'autre. Ces termes sont calculés à partir des taux de défaillance déterminés de manière conventionnelle pour chaque sous-ensemble sans redondance de la configuration.

L'objectif final est d'évaluer les probabilités des états 5 et 6 à chaque instant, ce qui donne la sécurité $S(t)$ et la disponibilité $A(t)$ en fonction du temps :

$$S(t) = 1 - P_6(t)$$

$$A(t) = 1 - P_5(t) - P_6(t)$$

Les valeurs numériques pour les taux de pannes sont extraits de la MIL HDBK 317B.

Les résultats sont globalement, pour la configuration examinée :

- en insécurité (taux d'erreurs indétectées) : 10^{-10} par heure pendant toute la durée de fonctionnement,
- en fiabilité globale pour la mission (sécurité et disponibilité) : 94% sur 5 ans pour une version spatiale.

GENERALITES SUR LE LOGICIEL

La machine virtuelle et le logiciel de base COPRA sont conçus pour satisfaire au mieux les exigences des traitements temps réel multitâches hiérarchisés demandant une haute sûreté de fonctionnement, une puissance de calcul élevée et une occupation mémoire réduite. Ils sont particulièrement bien adaptés aux systèmes embarqués sur avion ou satellite.

La tolérance aux pannes matérielles est assurée au moyen de procédures microprogrammées, d'instructions spécialisées et de procédures programmées faisant partie du logiciel de base qui complètent les dispositifs câblés de détections et de recouvrement de pannes. Ces fonctions étant assurées de façon transparente à l'utilisateur, l'analyse et la programmation peuvent être effectuées sans se préoccuper de cet aspect de la machine. Seuls les délais qui peuvent être induits par les recouvrements de pannes sont à prendre en considération pour les tâches temps réel. Ces délais sont spécifiés quantitativement de manière précise.

Les points de reprise sont injectés automatiquement à la génération des programmes, libérant ainsi le programmeur de leur gestion. L'algorithme d'insertion des points de reprise assure le découpage des programmes en

séquences répétées, tout en optimisant le nombre de points de reprise injectés.

La structuration de la programmation garantit la fiabilité du logiciel utilisateur en assurant le cloisonnement de l'information en mémoire (figure 4), la normalisation des interfaces logiciels, ainsi que le recouvrement des erreurs logicielles détectées conformément aux spécifications de l'application.

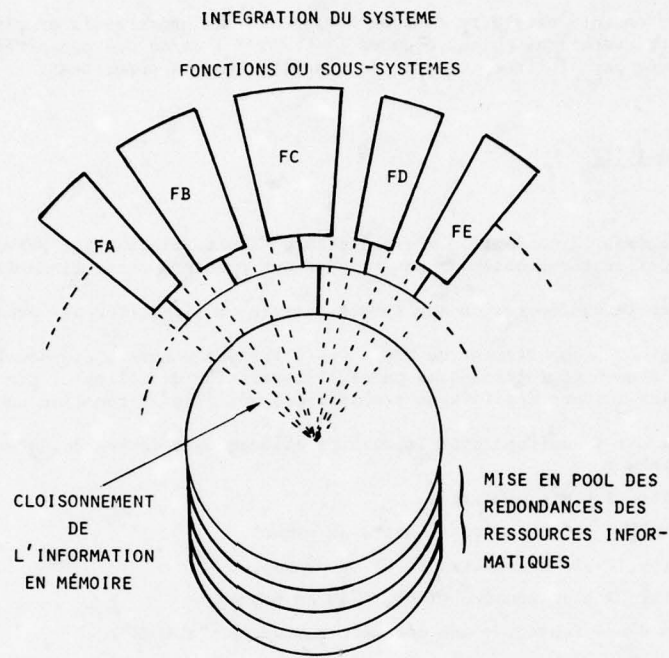


FIGURE 4

Le cloisonnement de l'information en mémoire est assuré par un contrôle strict de l'adressage des données. Une partie de ce contrôle est effectuée statiquement à la génération des programmes. Le reste (cas des adresses calculées) est effectué dynamiquement (par microprogrammation), sans pénalisation des performances. Les droits d'accès de chacune des tâches doivent être spécifiés explicitement afin de permettre ces contrôles.

La gestion des zones de travail temporaires est assurée automatiquement, ainsi que les sauvegardes et restaurations de contexte lors des commutations de tâches.

Deux modes de programmation sont possibles : "utilisateur" et "système de base". Les instructions spécifiques "système" ne sont utilisables que sur présentation d'un mot de passe.

En mode "utilisateur", les instructions "système" sont manipulables à travers un jeu complet de macro-instructions.

La puissance de calcul est le résultat d'un jeu d'instructions optimisé pour les domaines d'application du COPRA. Elle est encore augmentée par le fonctionnement possible en multiprocesseur.

En particulier, les techniques d'adressage par rapport à des bases spécialisées permettent d'accéder avec des instructions en format court (16 bits) à la quasi-totalité des objets adressés. La gestion des registres de base est automatique et transparente à l'utilisateur.

Les instructions de contrôle (branchements, appels et retours de sous-programmes, commutations de tâches, primitives de synchronisation) sont simples et puissantes.

L'allocation dynamique des espaces de travail temporaires assure la réentrance du code et va dans le sens d'une occupation mémoire réduite.

Enfin, il est possible de spécialiser le langage source par macro-instructions et fonctions spéciales micro-programmées (extra-codes) afin de l'adapter aux besoins particuliers de l'application.

STRUCTURATION DE LA PROGRAMMATION (figure 5)

L'application complète peut se décomposer en fonctions distinctes et relativement indépendantes, coopérant à l'accomplissement de la mission.

DECOUPAGE FONCTIONNEL DE L'APPLICATION

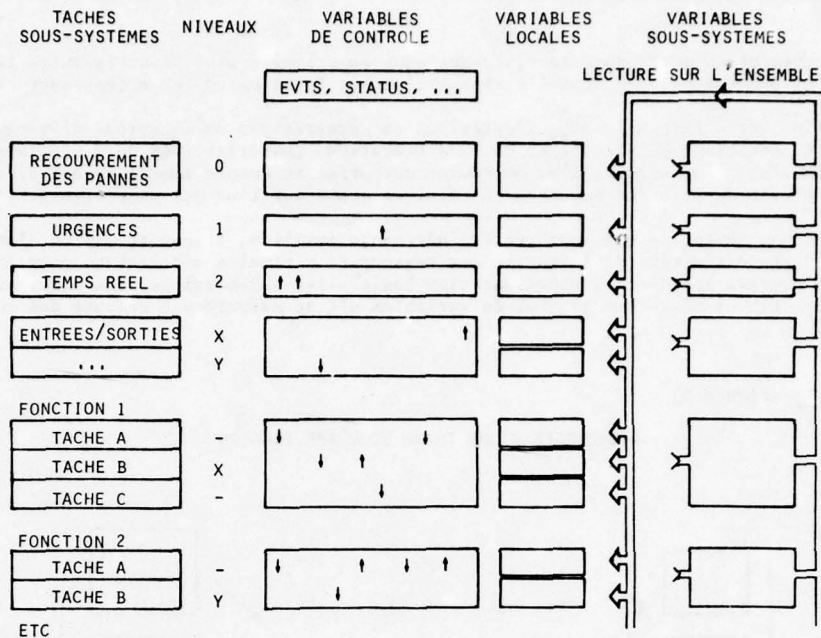


FIGURE 5

Une fonction peut elle-même se décomposer en plusieurs tâches correspondant à des conditions d'activation variées (différentes sous-fréquences du temps réel, interruptions aléatoires, événements externes ou internes).

Nous appelons sous-système, l'ensemble des tâches contribuant à assurer une même fonction. L'ensemble des variables communes aux tâches d'un même sous-système forme un espace de variables, dit espace global du sous-système. Seules les tâches du sous-système peuvent écrire dans cet espace, mais toutes les tâches de l'application peuvent y lire. Les espaces globaux sont alloués statiquement et les variables qu'ils contiennent peuvent donc être rémanentes.

Chaque tâche dispose aussi d'un espace de travail privé qui n'est adressable que par elle.

Nous appelons processus l'ensemble du code caractérisant une tâche (instructions). La notion de processus est une notion statique alors que la tâche est une notion dynamique. En particulier, plusieurs tâches peuvent se dérouler simultanément sur le même processus avec des zones de travail privées distinctes.

Un processus peut être décomposé en blocs emboîtés selon une structure arborescente. Un bloc est une séquence de codes ayant un ou plusieurs points d'entrée et de sortie, l'entrée s'effectuant par une instruction d'appel, la sortie effectuant toujours un retour derrière l'ordre d'appel. Chaque bloc dispose d'un espace de variables propres, dit espace local. Les espaces locaux sont gérés en pile au sein de l'espace de travail de la tâche.

La gestion en pile des zones locales permet d'optimiser la taille mémoire nécessaire à l'exécution de la tâche. L'espace local d'un bloc est alloué en sommet de pile lors de l'appel et libéré au retour. Il a donc la durée de vie du bloc. Ainsi, seule la zone locale du bloc racine de l'arbre des blocs peut contenir des variables rémanentes, les autres zones locales ne pouvant être que des zones de travail temporaires.

Les sous-programmes sont également structurés en blocs emboîtés. Ils peuvent être propres à un processus ou regroupés au sein de bibliothèques utilisables par tous les processus du système.

Le transfert du contrôle à un bloc ou un sous-programme est effectué par une macro-instruction normalisée assurant le transfert des paramètres et la création d'un lien de retour. La portée des instructions de branchement (GOTO) est limitée au bloc courant, afin d'éviter une entrée ou une sortie de bloc non normalisée et une utilisation anarchique de la pile de travail de la tâche.

GESTION DES TACHES

Une grande partie du système de gestion des tâches est microprogrammée de façon à la rendre particulièrement rapide. En configuration standard, il peut supporter 64 tâches réparties en 16 niveaux de priorité (y compris les niveaux d'interruption qui ne bénéficient pas d'un traitement spécial).

Un scheduler microprogrammé assure la sélection du niveau le plus prioritaire et active soit directement une tâche utilisateur si celle-ci est unique à ce niveau de priorité, soit un moniteur programmé de niveau si celui-ci abrite plusieurs tâches. Le moniteur de niveau utilise une instruction microprogrammée de commutation de contexte. Il peut assurer une gestion hiérarchisée ou non des tâches du niveau, selon les besoins.

Lors du fonctionnement en multiprocesseur, les niveaux de priorité sont répartis entre les processeurs. En cas de perte d'un processeur, une nouvelle répartition est faite entre les processeurs restants.

La synchronisation entre tâches et avec l'extérieur est assurée par un mécanisme d'événements. Ils peuvent être définis à l'assemblage ou alloués en cours d'exécution. Les primitives de manipulation des événements sont microprogrammées. L'apparition d'un événement est prise en compte immédiatement si elle a pour effet de relancer un niveau de priorité supérieur à celui en cours sur l'un des processeurs.

Sur apparition d'une faute logicielle détectée (adressage invalide, time-out,...) ou d'une panne matérielle entraînant une perte définitive d'une partie des ressources minimales nécessaires pour assurer l'ensemble de la mission, une dégradation douce (reconfiguration logicielle) est possible, aux mains de l'utilisateur. Il est utilisé pour cela un mécanisme général de variables d'état associées à chacune des ressources matérielles ou logicielles.

MACHINE VIRTUELLE (figure 6)

RESSOURCES D'UNE TACHE DE L'APPLICATION

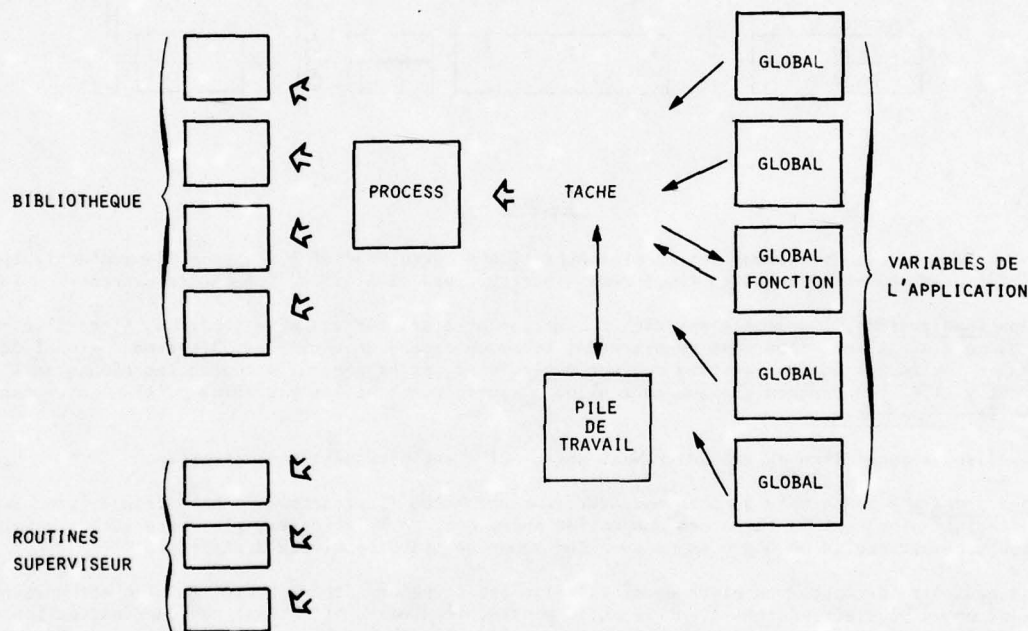


FIGURE 6

Vue de l'utilisateur, la mémoire se présente comme un bloc de 64K mots de 16 bits découpé en segments de longueur quelconque. L'adressage y est en général du type base + déplacement (éventuellement + index).

Les instructions et les constantes sont regroupées en segments CODE dans lesquels l'écriture est interdite, et qui peuvent être implantés en mémoire morte. Deux types de segments CODE sont différenciés, segment PROCESS qui contient un processus et ses sous-programmes propres et segment LIBRARY qui contient une bibliothèque de sous-programmes.

Les variables sont regroupées en deux types de segments : segment GLOBAL qui contient la zone globale d'un sous-système et segment STACK qui contient la zone de travail d'une tâche.

La machine virtuelle possède quatre registres de travail de 16 bits, dont deux peuvent être utilisés en index ou en pointeurs pour l'adressage externe (zones globales des autres sous-systèmes) et quatre registres spécialisés : base globale pointant sur l'origine de la zone globale, base locale pointant sur le sommet de la pile des zones locales de la tâche en cours, compteur ordinal et PSW. Les opérandes standard sont de 16 bits (virgule fixe) ou de 32 bits (virgule fixe ou flottante). Dans ce cas, on utilise deux ou quatre registres de travail de 16 bits pour faire un ou deux registres de 32 bits. Il est aussi possible d'adresser l'octet.

Les modes d'adressage autorisés à l'utilisateur sont les suivants :

- en zone globale : déplacement sur la base globale avec possibilité d'indexation,

- en zone locale : déplacement sur la base locale sans indexation (car la pile de travail contient des liens qui doivent être protégés),
- pour les paramètres des blocs et sous-programmes : adressage indirect spécialisé avec possibilité de post-indexation,
- par pointeur : déplacement sur un des registres d'index utilisé en pointeur, avec possibilité d'indexation avec l'autre registre d'index,
- en absolu : avec possibilité d'indexation.

L'adressage indirect banalisé est interdit car il ne peut être surveillé de façon efficace. Dans le cas des paramètres, la transmission est assurée de façon normalisée par macro-instruction et surveillée à la génération des programmes. L'adressage indexé est surveillé par la microprogrammation qui vérifie qu'il n'y a pas débordement du segment référencé. Les adressages externes (par pointeur ou en absolu) bénéficient d'un double contrôle : vérification statique du droit d'accès et vérification dynamique de l'adressage.

BAIE DE GENERATION DES PROGRAMMES ET MISE AU POINT

Les fonctions de panneau avant, de baie de mise en oeuvre, de baie de préparation de programmes et de mise au point sont assurées par un mini-ordinateur standard connecté au bus du COPRA.

CERTIFICATION

Il n'existe malheureusement pas de technique permettant de chiffrer la fiabilité d'un logiciel et c'est pourquoi la validation des mécanismes logiciels décrits précédemment sera l'objet d'une campagne d'essais et de certification effectuée par le CELAR fin 1980. L'accent sera mis sur les deux points principaux :

- cloisonnement garanti de l'information en mémoire entre sous-systèmes,
- survivance aux erreurs logicielles détectées par dégradation douce de la mission.

ETAT ACTUEL ET CALENDRIER DES DEVELOPPEMENTS EN COURS

Comme il a été dit en introduction, une maquette probatoire a été réalisée et expérimentée en 1976.

Les essais suivants ont notamment été exécutés :

- injection de pannes permanentes pré-câblées (court-circuits, collage...),
- injection de pannes transitoires aléatoires au moyen de sondes inductives,
- injection de pannes permanentes aléatoires par déconnexion de circuits ou de cartes, ou par production de courts-circuits.

La maquette a bien résisté à ces conditions d'environnement défavorables.

L'étape suivante en cours est celle d'un prototype à basse consommation destinée aux applications spatiales.

Pour cela, une version équivalente de l'AMD 2901 a été réalisée en CMOS sur SOS par la société EFCIS en collaboration avec la SAGEM. Ce circuit "ALI" (Arithmétique et Logique Intégrée) dont les échantillons ont démontré un niveau de performance sous 10V comparable à celui du 2901A pour une dissipation 10 fois moindre, doit être industrialisé au cours de l'année prochaine. Ce circuit devrait être complété par un micro-séquenceur cascable dans la même technologie afin de permettre la réalisation complète d'un processeur en CMOS et CMOS sur SOS travaillant sous 10V (vitesse et immunité au bruit maxima).

Les mémoires de microprogrammes sont des ROM en CMOS de 1K x 12 bits. Les RAM sont en CMOS de 1K x 4 bits. La puissance de calcul offerte sera alors de 400 kop/s (ou 200 kop/s avec un seul processeur allumé), et la consommation de 15W sur le réseau 28V continu de bord.

Dans l'intervalle, le processeur du prototype sera en grande partie monté avec des circuits bipolaires. Une étape ultérieure prévoit la connexion de l'enregistreur reconfigurable à bulles magnétiques de 60 M bits en cours de développement pour le compte de l'ESA (Agence Spatiale Européenne).

Le prototype en développement doit être livré au CELAR (Centre d'Essais des Laboratoires de l'Armement) au cours du deuxième semestre 1980. Il subira alors une campagne d'essais de certification au niveau matériel et logiciel.

CONCLUSION

L'accent principal est mis actuellement vers les applications pour satellite pour lesquelles les fiabilités demandées atteignent 95% sur au moins cinq ans.

D'autres applications envisagées concernent la conduite des systèmes intégrés sophistiqués des véhicules militaires futurs.

Pour toutes ces applications, une configuration appropriée de la famille COPRA peut offrir une réponse adaptée à chaque cas particulier en terme de puissance de calcul, de sécurité et de disponibilité.

Concernant l'aspect logiciel de la fiabilité, pour lequel il n'existe pas aujourd'hui de réponse complète et satisfaisante, la structuration imposée à la programmation et le cloisonnement étroitement surveillé des divers objets manipulés par le logiciel, enfin les outils offerts pour la conduite d'une dégradation douce, devraient permettre une amélioration notable des conditions de certification des logiciels d'application.

Dans tous les cas, l'utilisation de la structure COPRA apporte au départ l'assurance du niveau de sûreté de fonctionnement requis pour l'unité centrale en toute indépendance des programmes d'application. Enfin, il libère ces derniers de toute sujétion concernant la survie aux pannes de l'unité centrale ce qui, en simplifiant le logiciel, améliore sa fiabilité.

COPRA

A NEW LINE OF ULTRA-RELIABLE, RECONFIGURABLE COMPUTERS
FOR AIRBORNE AEROSPACE APPLICATIONS

C. MERAUD AND F. BROWAEYS

SAGEM - 6, AVENUE D'IEA, 75783 PARIS CEDEX 16 - FRANCE

SUMMARY

The COPRA (Calculateur à Organisation Parallèle et à Reconfiguration Automatique / Automatic Reconfiguration and Parallel Organisation Computer) covers a range of equipment, from the redundant mono-processor (200 kops/sec) to the reconfigurable multi-processors (400, 600 and 800 kops/sec).

A brass-board has already been built and a prototype is being developed. The latter corresponds to a double-duplex configuration capable of surviving at least to a first failure and to transient failures in a manner transparent to the programming, which considerably simplifies the development of the application software.

The hard-wired fault detection, the automatic micro-programmed roll-back and the software reconfiguration mechanism are described. Some technological aspects, in particular the use of CMOS on SOS, are set forth.

The software is structured and rigorously segregated even at machine level. Tools are provided at supervisor level, to facilitate the programming of a graceful degradation of the mission.

A Markovian modelization enables an accurate estimation of reliability which, where the developed configuration is concerned, will be close to 95% over 5 years with a rate of undetected errors equal to 10^{-10} per hour.

INTRODUCTION

The conventional central units can be used to build process control systems of ever increasing complexity, as long as the reliability level required is not too high. In applications with non dangerous configuration (e.g. : telephone multiplexer), the availability may be improved by duplicating the central unit and, if possible, keeping the overall system under human supervision. Beyond this, it is necessary to use FTC (Fault Tolerant Computer) techniques. This is the case, in particular, for the built-in control systems on modern aircraft (safety requirements exceeding 10^{-10} per hour) and satellite-borne computers for which an availability of more than 95% over several years is demanded.

The family of COPRA computers under development is intended for such applications. It uses the most up-to-date FTC techniques which involve : the provision of error and fault detecting devices, recovery through reconfiguration, automatic roll-back of interrupted processes and, finally, the generation of reliable software.

In order to ensure the specified reliability on the most critical tasks, it is indeed necessary :

- to insert redundancies which will ensure computer survival and guarantee the integrity of the memorized information,
- ensure the safety of the computing operations (without increasing the complexity of the software), in order to prevent the production of catastrophic events.

The COPRA central unit meets these requirements and enables the reliable programming of important software. If a fault appears, it is detected and confined before the information is irreversibly polluted, as this would prevent the correct roll-back of the operations. These last operations are therefore performed with separate hardware, to prevent the defects from altering the process under way.

Since transient failures are the most frequent (99% of program failures), they are filtered at the hardware level, by an automatic roll-back of the sequence in progress.

If the fault is confirmed, the module which is at the origin of the fault is isolated, and the emergency module is initiated. An automatic process roll-back is then performed.

These operations are carried out simply, with the help of the microprogramming technique used by SAGEM since 1967 for all of its central units. This technique simultaneously offers :

- a list of instructions suitable for the domain of application concerned,
- some diagnostic instructions required for the execution of periodic tests which effectively eliminate hidden failures,
- some logic operations, specific to failure processing and recovery.

The improved reliability of the software is based on split tasks and a memorized information arrangement which is the image of the functional structure of the application. The guiding principles of communication and access between the various parts can then be described, to enable the static and dynamic verification of their observation.

It is thus possible to perform an efficient and monitored data segregation for each sub-system. It is also possible to provide a finer division into monitored functional tasks, to take full advantage of the possibilities offered by a graceful degradation of the application.

These characteristics restore a clear segregation of functions and sub-functions, fully comparable with what is conventionally performed in discrete component type redundant analog structures. The absence of such a segregation in conventional computers largely explains the reticence felt towards data processing, even when it is reliably protected against hardware failures.

In conclusion, due to its highly reliable structure and software characteristics, the FTC COPRA computer enables the satisfactory writing of application software since computer stoppage or faults need not be taken into account. Fault detection and automatic reconfiguration protect the mission against the consequences of computer stoppage or anarchic operation and make it possible to reach, in a rigorously assessable manner, the level of reliability required for the whole of the application.

All the points summarized in this introduction will be developed hereafter.

HISTORY

The development of COPRA is the fruit of a 6-year collaboration between the SAGEM Company (main contractor) and the Electronique Marcel Dassault Company. Both these Companies have been designing and building real time computers since the early 60's. They both tried to improve the functional reliability of computers by the use of redundancies at a time when the level of integration did not exceed the size of a register (SAPHIR and MECRA projects).

The end of the 60's saw the emergence of LSI techniques which led to a concept of more global redundancy, with a higher reliability combined with reasonable size and cost.

It is within this context that the two Companies combine their effort on the COPRA project, with the help of the French Department of Defense (via the DRET - Direction des Recherches, Etudes et Techniques).

In 1976, this effort led to the realization of a model used to validate the basic solutions : this is now followed by the construction of a prototype.

SURVIVAL TO FAILURES

1. Survival to transient failures (figure 1)

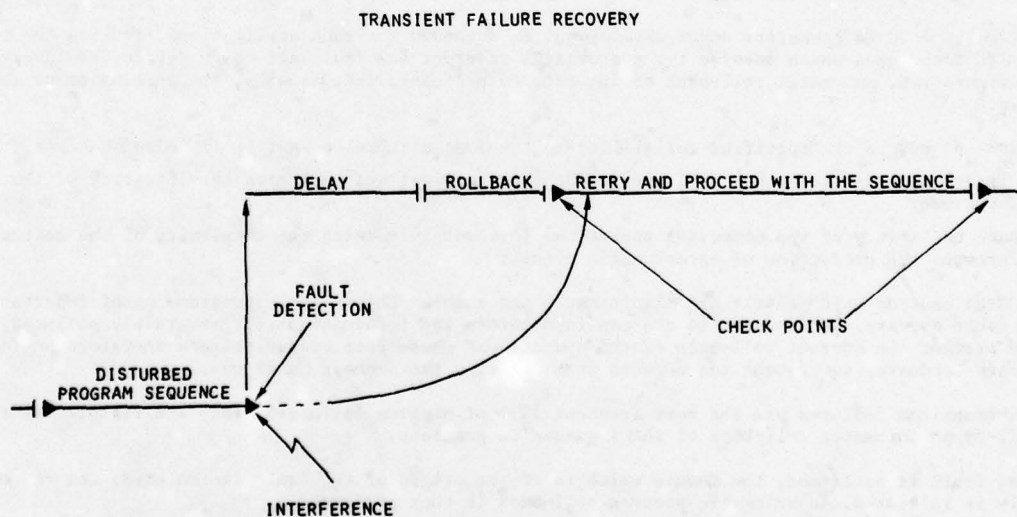


FIGURE 1

This survival, and the discrimination between this type of failure and solid failures in components, is one of the most important points in a system which detects any error with very high effectiveness.

It is indeed admitted that 99% of observed program defects or failures are due to transients produced by the environment or by components which, in some data configurations, are operating in marginal conditions.

In fact, this proportion would be even greater on conventional machines if all the errors affected some critical information, which is far from being the case. Therefore, an FTC machine which ensures safety by trying to detect all errors must be able to recognize the transient characteristic of their cause and survive without initiating a premature reconfiguration.

One solution consists in using a triplication and voting process : however, this solution, which in fact is complex, is vulnerable to dazzling and ill suited to long missions.

The only alternative is to make use of the sequential characteristic of digital processings, in order to introduce an automatic roll-back mechanism which, following the detection of an error on the sequence in progress, enables the retry of the latter from a previous check point.

Obviously, such mechanism must be transparent to the programming, in other words, it must not impose any restriction to the user.

To be simple and fast, without requiring too much additional equipment, the roll-back mechanism is obtained through micro-programming. It consists in performing, at each check point "CHK", a short protection of the context in the memory, in a manner similar to that performed at the time of an interrupt. The reverse operation, carried out after the detection of an error, performs a roll-back.

In such a mechanism, the check points cannot be located anywhere. Indeed, since the roll-back is carried out with the states in which the memory and I/O's are when the error is detected, these states must be compatible with the identical repetition of the same computations (E.g. : $A + B \rightarrow R, R \rightarrow A$ is not repeatable).

The location of the check points is effected during program generation, by means of a special algorithm. The latter determines from which supplementary action the sequence being assembled is likely to modify its initial conditions and then inserts a check point (E.g. : $A + B \rightarrow R$, "CHK", $R \rightarrow A$. The division into repeatable sequences is realized).

The check point is a point of no return and the start of a new sequence. The period of sensitivity is equal to a few nano-seconds : it is therefore almost nil. The average distance between check points varies between 10 and 20 instructions. They are previously set by a protection of processor registers in two zones working in flip-flop mode, the validation is effected by a switch over of zone. The setting operations have the duration of a fast instruction leading to a consumption of computing power equal to approximately 8%.

Following an error detection, the roll-backs are delayed by a fraction of a ms according to environmental conditions, in order to improve the protection in the presence of trains of transients. Their number is counted and their frequency compared with thresholds, for maintenance purposes.

Such a mechanism does indeed free the programmer from any need to provide protective measures against such failures. Consequently, no human error is to be feared when the check points are inserted in the program. As a result, software certification is much enhanced.

2. Error detection in processors - processor reconfiguration

The CPU's are composed of 4 AMD 2901 type 4-bit slice microprocessors, or equivalent equipment made by the EFCIS Company (CMOS on SOS). The word and data path have a 16-bit width. The floating operations work on operands with 24 mantissa bits and 8 exponent bits.

Error detection

In the recent past, the processing units represented a large part in the cost of computers consequently, SAGEM and EMD worked out some solutions based on arithmetic codes to avoid the duplication and comparison of processing units for the detection of errors.

This solution was finally abandoned since it was not sufficiently effective to monitor all the operations carried out by a processing unit, and ill suited for present LSI components. Since the use of the latter have brought down to approximately 10% the share of the processing unit in the cost of a complete computer, it was more economical to revert to the solution originally dropped. The present solution therefore comprises two processing units operating in micro-synchronization and comparing all their outputs bit by bit at each micro-cycle. The comparison itself only requires few MSI components.

Reconfiguration

At the first stage, the error recovery assumes a transient failure. The latter is recovered by the "Delay and Roll-back" microprogrammed sequence (figure 1).

When the sequence still shows a processing unit error, the failure is considered to be solid (figure 2). This confirmation leads to the switching off of the PU and the latter is electrically isolated from the system.

This state causes a maximum level interrupt detected by another PU used to resume processings temporarily abandoned.

A new task distribution is performed with the help of a new set of level masks. The balance is progressively restored from the natural management operation of a multi-level common purpose multi-processor, the levels of which are statically shared out among the processors.

In such a case, the reduced computing power left by the loss of a processor must be sufficiently high to continue the complete fulfilment of the mission. In the opposite case, the suppression of non critical tasks is envisaged, to perform a graceful degradation of the mission.

SOLID FAILURE RECOVERY

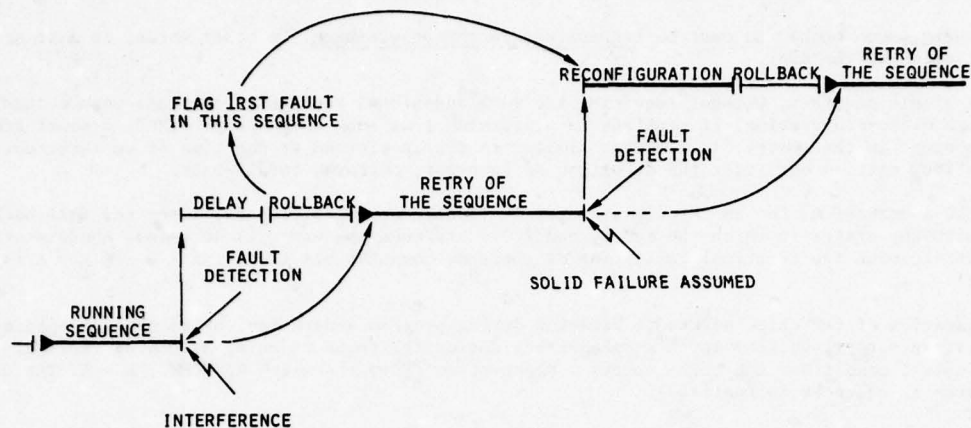


FIGURE 2

3. Error detection in memories - Memory reconfiguration

The memory is made up of CMOS integrated circuits (PROM, ROM, or RAM of various capacity with, however, a large majority of RAM 4K chips). These chips are grouped into 4 K pages, with a supplementary spare page. The integrity of the information, in the presence of failures, is ensured by a memorization duplicated in two physically independent units. Memory writing is simultaneous in both units. Reading is performed on only one unit at the time, which approximatively doubles the memory output. There must be at least three failures among the memory chips in order to lose the memory function.

Detection

Beside the memorization circuits which are code-monitored (as will be described hereafter), the memory service logic circuits (access dialogue, decoding, address, page reconfiguration) are, for the same reasons as described previously, monitored through duplication and comparison.

Any discrepancy detected at comparison level immediately inhibits transfers and causes a micro-interrupt on the processors, in order to execute the "Delay and Roll-back" microprogrammed sequence.

The memorization circuits which, when dealing with high capacities, may form a costly part of the system, are monitored through coding. To this end, the circuits are grouped in independent memory slices of 1 or 4 bits (even 8 bits). An additional slice is provided to accommodate the detecting code. With such an arrangement, component failures can produce an error in only one slice. Such error configurations are perfectly detected as long as the detection system itself is not faulty. A periodical test is carried out to provide against such a risk.

Reconfiguration

When an error is detected, it initiates the previously mentioned microprogrammed "Delay and Roll-back" sequence which recovers the error if the latter is of the transient type.

If the error persists, a microprogrammed correction is performed from the second copy of the information kept in the second memory unit. This operation filters the memorized transient failures.

If the fault is still present, the memory is then reconfigured.

This operation takes place in two stages :

- immediately, the processor in dialogue with the faulty memory unit sets a flip-flop which disables the faulty unit and prevents it from being used by the processors. The computing operations proceed with the information which, from then on, is available in single copy form in the remaining unit,
- later on, the periodic program of test and memory reconfiguration carries out the complete diagnostic and repair, with the help of specific microprogrammed instructions.

If the failure is located in a 4 K page (the most likely event) and if the spare page is still available, the latter is used in the place of the defective page. Then, the whole unit is up-dated through microprogrammed reproduction from the other unit, and the unit is re-introduced into the overall configuration.

If the failure configuration prohibits the overall use of the unit (failure of logic services, bus sticking, unit, ...), or if the spare page has failed or is already used, the provisional decision of non availability is definitively confirmed by a signal sent by the processor dealing with the fault, which switches off the supply to the unit.

Finally, if the second memory unit has already failed, or when the failure only causes the loss of a page containing non critical data, a graceful degradation is performed. If the lost page contains critical data, or if the fault disables the complete unit, the complete failure alarm is initiated.

At the present stage, it is possible to understand why the use of a correcting code is not possible to avoid the duplication of the memory. The amount of memory logic service failures is indeed not negligible. At this level, it would be necessary to provide an additional redundancy simultaneously prewired to all 4-bit slices of the memory and to all processors. Such an arrangement considerably increases the complexity of the interfaces and reduces the independence of failure causes between redundant sections. It is for these reasons that the use of self-correcting codes has been avoided.

4. Error detection in I/O - Reconfiguration of I/O

At central unit level, the inputs/outputs are made with controllers connected to the internal buses of the memory units. They are therefore duplicated. With regard to programming, they appear as transfer operations in a specific page.

The links with sub-systems are by means of a duplicated multiplexed serial bus on the buses of each memory unit. They are controlled by a microprogrammed channel present in each processor. The channel function therefore has fault detection and reconfiguration characteristics specific to the processor function.

With regard to central unit failures, and up to I/O controller internal failures, it will be assumed that the application of the above mentioned principles ensures the survival of the I/O function to numerous CU failures and, at least, to any first failure configuration.

Beyond this, to survive to sensor, activator, display failures, etc..., the COPRA structure provides the user with the means to connect such devices via two independent path and to be informed at software level of the application of any detected anomaly. Not having to worry about a failure of the CU, the user can thus make a judicious choice with regard to the location of redundancies within the whole of the application, and make the best use of the application context data to obtain the required I/O reliability.

At software level, the user is provided with the means required to elaborate an effective programming of graceful application degradation.

5. Other functions

All supplies are duplicated and only one is in use at a given time.

The clocks, with their already high intrinsic reliability, are triplicated and voted at the input of each function.

There is no crossing point on the buses interconnecting the various redundant functions which includes components with a significant rate of failure.

It has been proved that no first failure of components wired on buses (short-circuit, sticking, etc...) or on function access circuits, could lead to the loss of more than one processor and one memory unit at the time, this last failure configuration being itself highly unlikely.

A FAMILY OF ULTRA-RELIABLE COMPUTERS

Using the basic modules previously described, it is possible to configure the following computers :

1. Elementary mono-processor (UT 382-100)

The elementary mono-processor, built round four AMD 2901 slice type microprocessors or their ALIS (SOS) equivalent, presents the following characteristics :

- multi-layer PCB packaging compatible with short 1/2 ATR (1 MCU) dimensions or 3/4 ATR (6 MCU) transversal dimensions,
- 350 kop/s output,
- 150 basic instructions and provision for specific instructions,
- 48-bit microprogramming,
- fixed operations on 16 or 32 bits. Floating operations on 24 mantissa bits and 8 exponent bits.

The memory comprises 1 to 4 1K RAM pages associated with 1 to 10 2K EPROM pages, making a total of 24K words on a single board with a format identical to that of the processor. This capacity may be extended on additional boards (up to a maximum theoretical number of 256K words).

This machine is not fault-tolerant.

2. Transient fault-tolerant basic mono-processor (UT 382-200)

To the previously described basic configuration are added :

- a second processor with monitoring circuits, to make up the self-monitored UT 382-200 processor : this "passivated failures" type processor is composed of two adjacent PCB's (with possible electromagnetic shielding),

- the previous memory is completed with its monitoring and error detection bit storage circuits.

The microprogrammed roll-back mechanism is provided, allowing this machine to withstand transient failures.

3. Fault-tolerant bi-processor under development

This is a typical "double-duplex" organization (figure 3), ensuring maximum reliability and a very high availability.

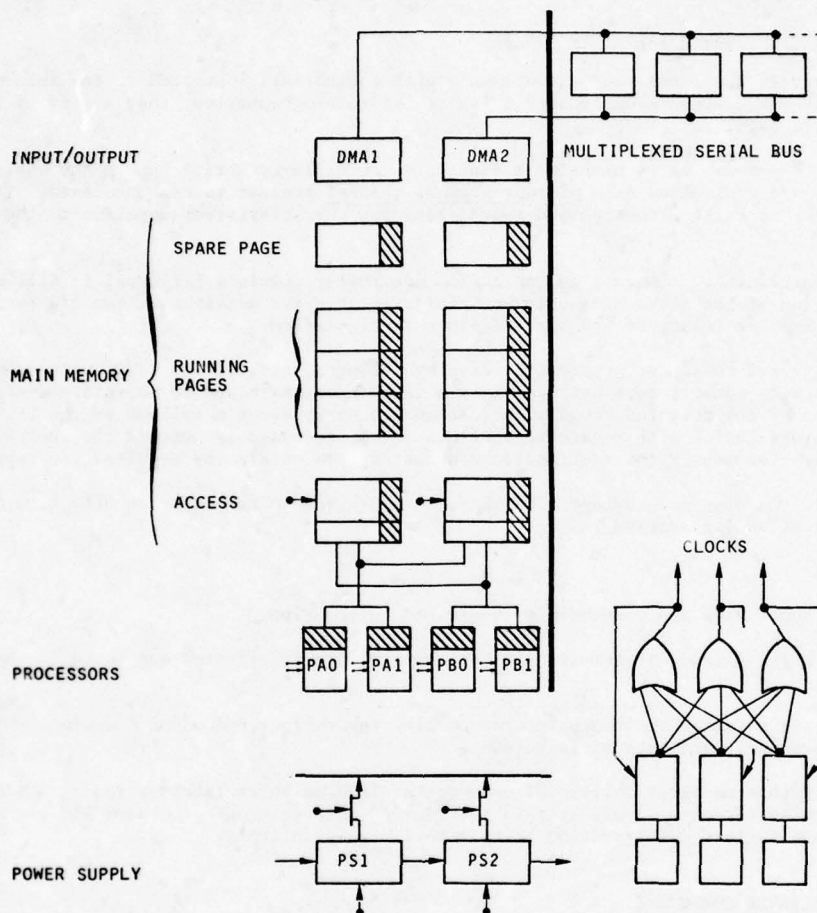


FIGURE 3

It comprises two duplex processors, two self-monitored memory units, two duplex I/O controllers, a TMR clock and duplicated services.

With the two processors ON, the on-line power can reach 600 kop/s (T2L version) or 400 kop/s (CMOS on SOS version). For space application, one of the processors may be switched OFF, to reduce power consumption and improve reliability: the speed is then equal to 200 kop/s.

The memory is composed of two identical units on one PCB, each unit being similar to the one used in the UT 382-200.

The I/O include a duplicated multiplexed serial bus (already described) and some discrete I/O.

The assembly is compatible with the dimensions of a short 1/2 ATR.

In the space version (CMOS on SOS), the power consumption with one processor ON (200 kop/s) and an effective capacity of 16 K, is equal to 15 W (from the airborne 28 V DC supply).

4. High memory capacity multi-processor configuration

The previously described organization may be extended to more than two processors and more than two memory PCB's, to obtain high computing powers (800 kop/s) and/or high memory capacities, provided that the internal parallelism possibilities of the application allow it.

EVALUATION OF RELIABILITY

In fault-tolerant architecture design, it is essential to provide a modelization in order to quantify the improvements obtained as compared with conventional non redundant machines.

This makes it possible to adjust the configuration to the reliability specifications laid down for each specific application.

To this end, a COPRA behaviour (relative to failures) model has been developed, using the Markov processes. Starting with an initial failure-free state, the appearance of failures causes transitions towards various possible states, according to the laws of probability defined as a function of time.

A simplified example is given hereunder, for the COPRA double-duplex configuration. This configuration may be in the following states :

- state 1 : 2 duplex UT, 2 memory units,
- state 2 : 2 duplex UT, 1 memory unit (the other unit has failed),
- state 3 : 1 duplex UT, 2 memory units (1 UT failed),
- state 4 : 1 duplex UT, 1 memory unit (1 UT and 1 MU failed),
- state 5 : fault detected on one last ressource ("ALARM"),
- state 6 : fault undetected at any given stage.

If $\Pi(t)$ is the line vector ($P_1(t), \dots, P_6(t)$) of the probabilities of having reached each of the states defined at instant (t), the evolution of that vector satisfies the following differential equation :

$$\frac{d \Pi(t)}{dt} = \Pi(t) \cdot \Lambda(t)$$

where $\Lambda(t)$ is a 6-dimension square matrix the terms of which are the rates associated with the probabilities of transition from one state to the other. These terms are computed from failure rates determined in conventional manner for each sub-assembly, without configuration redundancy.

The ultimate object is to evaluate the probabilities of states 5 and 6 at each instant, which gives safety $S(t)$ and availability $A(t)$ in time :

$$S(t) = 1 - P_6(t)$$

$$A(t) = 1 - P_5(t) - P_6(t).$$

The failure rate numerical values are taken from MIL HDBK 317B.

With the configuration examined, the overall results are as follows :

- unsafe state (rate of undetected errors) : 10^{-10} per hour, during the complete operation,
- overall reliability for the mission (safety and availability) : 94% over 5 years, for the space version.

SOFTWARE - GENERAL

The virtual machine and the basic COPRA software are designed to satisfy, to the best possible extent, the requirements of hierarchical multi-task real time processings demanding high functional reliability, high computing power, and minimum memory space. They are particularly well suited for aircraft or satellite-borne systems.

The tolerance to hardware failures is obtained by means of microprogrammed procedures, special instructions and programmed procedures included in the basic software, which complete the wired failure detection and recovery devices. Since these functions are transparent to the user, the analysis and programming may be carried out without taking this aspect of the machine in consideration.

The delays which may be induced by failure recovery are, alone, to be taken into consideration for real time tasks. These delays are specified quantitatively, in a precise manner.

The check points are injected automatically when the programs are generated, thus relieving the programmer of this task. The check point insertion algorithm ensures the splitting of programs into repeatable sequences, while optimizing the number of check points injected.

The structure of the program guarantees the reliability of the user software by ensuring the segregation of data in main memory (figure 4), the standardization of the software interfaces, and the recovery of detected software errors in accordance with the specifications of the application.

The segregation of the information in main memory is achieved through strict monitoring of data addressing. Part of this monitoring is carried out statically when the programs are generated. The remainder (computed addressings) is carried out dynamically (through microprogramming) without penalizing the performance. The rights of access of each task must be explicitly specified in order to enable such monitoring operations.

The control of temporary working zones, as well as the protection and the restoration of context during task switching, are carried out automatically.

SYSTEM INTEGRATION

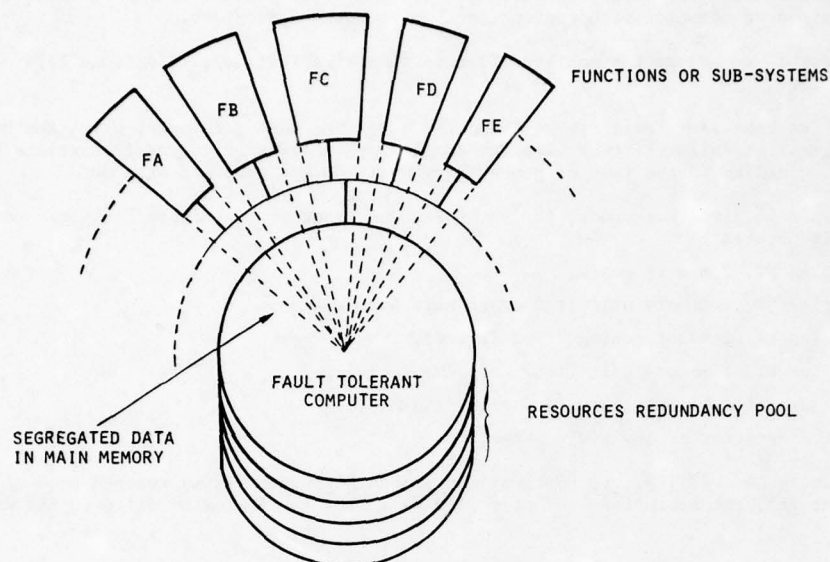


FIGURE 4

Two programming modes are possible : the "user" and the "basic system" modes. The specific "system" instructions can be used only when a pass-word is presented at program generation.

In the "user" mode, the "system" instructions can be handled through a complete set of macro-instructions.

The computing power results from a set of instructions optimized for the domains of application of the COPRA. It is further increased by a possible operation in multi-processor mode.

In particular, the addressing techniques relative to specialized bases provide access to practically all addressed objects with the help of short format (16 bits) instructions. The control of the base registers is automatic and transparent to the user.

The monitoring instructions (branches, sub-program calls and returns, task switching, synchronization primitives) are simple and powerful.

The dynamic assignment of temporary working spaces ensures the re-entrance of the code and helps reducing the amount of memory space occupied.

Finally, it is possible to specialize the source language through macro-instructions and special micro-programmed functions (extra-code), to match it to the specific needs of the application.

PROGRAM STRUCTURE (figure 5)

The complete application may be splitted into distinct and relatively independent functions, cooperating to the fulfilment of the mission.

A function can itself be splitted into several tasks corresponding to various activating conditions (different sub-frequencies of real time, random interrupts, external or internal events).

We call sub-system, the set of tasks which contribute to the fulfilment of a same function. The group of variables common to the tasks of the same sub-system forms a space of variables called the sub-system global space. The sub-system tasks alone can write in that space but all the tasks of the application can read in it. The global spaces are assigned statically and the variables they contain can therefore be remanent.

Each task is also provided with a private working space which can be addressed only by it.

We call process, the code which characterizes a task (instructions). The notion of process is a static notion while the task is a dynamic notion. In particular, several tasks can be performed simultaneously on the same process, with distinct private working zones.

A process may be splitted into nested blocks according to an arborescent structure. A block is a sequence of codes with one, or more, points of entry and exit, the entry being initiated by a call instruction and the exit being always initiated by a return after the call order. Each block is provided with a space of

FUNCTIONAL SPLITTING UP OF THE APPLICATION

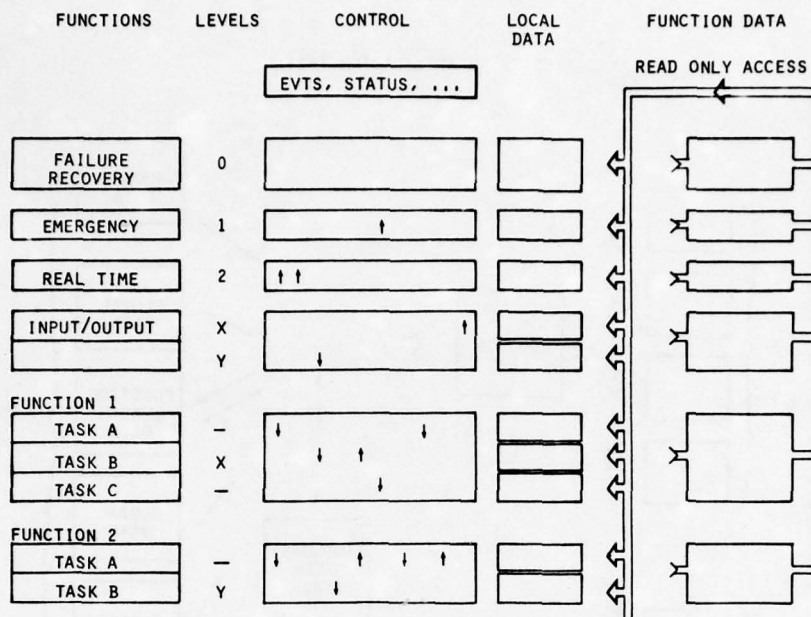


FIGURE 5

its own variables called the local space. The local space are managed in stack, within the working space of the task.

This stack management of local spaces makes it possible to optimize the size of the memory required to execute a task. The local space of a block is allocated to the top of the stack when a call is made and then released upon the return order. It lasts as long as the block lives. Thus, the local zone of the block which forms the root of the tree of blocks can, alone, contain remanent variables : the other local zones are merely temporary working zones.

The sub-programs are also structured into nested blocks. They may be specific to a process or grouped within libraries which can be used by all system processes.

The transfer of control to a block or sub-program is effected by a standardized macro-instruction ensuring the transfer of parameters and the creation of a return link. The scope of branching instructions (GOTO) is limited to the current block, to prevent a non standardized block entry or exit and the anarchic use of the task working stack.

TASK MANAGEMENT

A large part of the task management system is microprogrammed to make it particularly fast. In the standard configuration, it can support 64 tasks shared out into 16 levels of priority (including the interrupt levels which require no special treatment).

A microprogrammed scheduler ensures the selection of the level with the highest priority and activates either a user task if the latter is the only one at that level of priority, or a level programmed monitor if it accommodates several tasks. The level monitor uses a microprogrammed context switching instruction. As required, it can ensure the hierarchical, or non hierarchical management of the level tasks.

In the multi-processor operating mode, the priority levels are shared out among the processors. If one processor is disabled, they are shared out among the remaining processors.

The synchronization between tasks and with the outside is performed by an event mechanism. They may be defined at assembly, or allocated during the execution. The event handling primitives are microprogrammed. The presence of an event is immediately taken into account if its object is to reinstate a priority level higher than the one in progress on one of the processors.

Upon the appearance of a detected software fault (invalid addressing, time-out, ...), or a hardware failure leading to the definitive loss of part of the minimum resources necessary to fulfil the whole of the mission, the user can initiate a graceful degradation (software reconfiguration). This brings into use a general mechanism of state variables associated with each of the hardware or software resources.

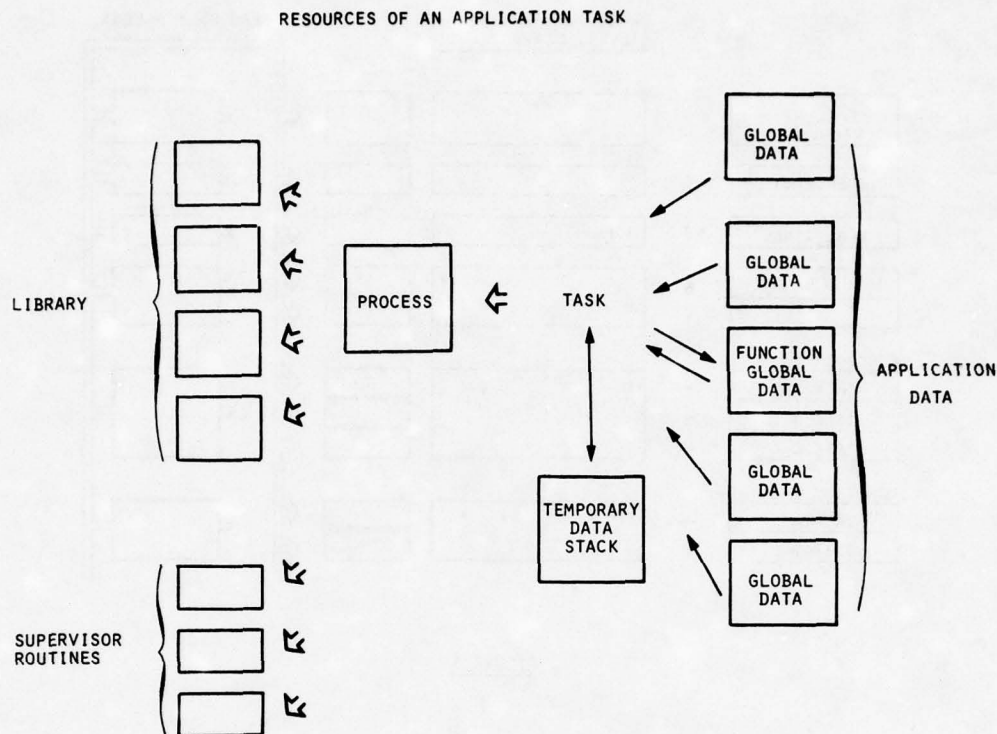
VIRTUAL MACHINE (figure 6)

FIGURE 6

Viewed by the user, the memory appears as a block of 64 K words of 16 bits, sliced into segments, each of any given length. Generally, the addressing is of the "base + displacement" type (possibly + index).

The instructions and constants are grouped into CODE segments in which writing is prohibited and which may be located in a read-only memory. Two types of CODE segments are differentiated : the PROCESS segment, containing a process and its own sub-programs, and the LIBRARY segment containing a library of sub-programs.

The variables are grouped into two types of segments : the GLOBAL segment, containing the global zone of a sub-system, and the STACK segment, containing the working zone of a task.

The virtual machine is provided with four 16-bit working registers, two of which may be used as indexes or pointers for external addressing (global zones of other sub-systems), and four specialized registers : global base pointing to the origin of the global zone, local base pointing to the top of the stack of local zones of the running task, program counter, and PSW. The standard operands have 16 bits (fixed point) or 32 bits (fixed or floating point). In this case, we use two or four 16-bit working registers to make up one or two 32-bit registers. It is also possible to address the byte.

The addressing modes which may be employed by the user are as follows :

- in global zone : displacement on the global zone, with a possibility to index,
- in local zone : displacement on local base with no indexing (because the work stack contains links which must be protected),
- for block and sub-program parameters : indirect specialized addressing with post-indexing possibility,
- by pointer : displacement on one of the index registers used as pointer, with a possibility to index with the other index register,
- absolute : with indexing possibility.

Common indirect addressing is prohibited because it cannot be effectively monitored. In the case of parameters, transmission is ensured in a standard manner by a macro-instruction and monitored when the programs are generated. Indexed addressing is monitored by the microprogramming which ensures that there is no overflow of the referenced segment. External addressing (by pointer, or absolute) benefit from a double check : static verification of access right and dynamic verification of the addressing.

PROGRAM GENERATION AND AIDS

The front panel, operating cabinet and program preparation and debugging functions are fulfilled by a mini-computer connected to the COPRA bus.

CERTIFICATION

There is, unfortunately, no technique which can be used to evaluate the reliability of a software. This is the reason why the validation of the previously described software mechanisms will be the object of a trial and certification campaign which will be carried out by CELAR (Centre d'Essais des Laboratoires de l'Armement) at the end of 1980. Particular attention will be paid to two main points :

- guaranteed segregation of the information in main memory between sub-systems,
- survival to detected software errors, through graceful degradation of the mission.

PRESENT SITUATION AND TARGET DATES FOR DEVELOPMENT

As already mentioned in the introduction, a brass-board has been built and experimented in 1976.

The following tests have, in particular, been carried out :

- injection of pre-wired solid faults (short-circuits, sticking, ...),
- injection of random transient faults by means of inductive probes,
- injection of random solid faults, through disconnection of circuits or PCB's or production of short-circuits.

The brass-board stood up satisfactorily to those adverse environmental conditions.

The next step in progress covers a low-consumption prototype intended for space applications.

To this end, a version equivalent to the AMD 2901 has been made in CMOS on SOS form by the EFCIS company, in collaboration with SAGEM. This "ALIS" circuit "Integrated Arithmetic and Logic on SOS", the samples of which have demonstrated a performance level (under 10 V) comparable to that of the 2901A, with 10 times less dissipation, is due to be industrialized during the next year. This circuit should be completed by a cascade micro-sequencer of identical technology, in order to enable the complete construction of a CMOS and CMOS on SOS processor operating under 10 V (max. speed and immunity to noise).

The microprogram memories are of the 1 K x 12 bits CMOS ROM type. The RAM are of the 1 K x 4 bits CMOS type. The available computing power will then be equal to 400 kop/sec (or 200 kop/sec with only processor switched ON) and the power consumption equal to 15 W (from the airborne 28 VDC supply).

In the meantime, the processor of the prototype will be largely based on bipolar circuits. A later step involves the connection of the 60 M bit-magnetic bubble reconfigurable recorder under development for the ESA (European Space Agency).

The prototype under development is due to be delivered to the CELAR during the second half of 1980. It will then be subjected to a certification tests campaign (software and hardware).

CONCLUSION

At present, the emphasis is placed on satellite applications for which the reliability requirements reach 95 % over at least 5 years.

Other applications envisaged include the control of sophisticated integrated systems intended for future military vehicles.

For all these applications, an appropriate configuration of the COPRA family can offer, in terms of computing power, reliability and availability, a suitable solution to each specific case.

With regard to the software aspect of reliability, for which no complete and satisfactory solution is yet available, the imposed programming structuration and the closely supervised segregation of the various objects handled by the software, as well as the instruments provided to conduct a graceful degradation, should allow a substantial improvement of the application software certification conditions.

In all cases, the use of the COPRA structure ensures, from the start, the functional reliability level required for the central unit, independently of the application programs. Finally, it frees the latter from any subjection concerning the CPU survival to failures : as a result, the software is simplified and, therefore, more reliable.

A HIGH ACCURACY FLIGHT PROFILE DETERMINING SYSTEM

by

Peter Roy Vousden, B.Sc., C.Eng.
and

Dr. Peter Jonathon Gollop

LITTON SYSTEMS CANADA LIMITED
25 Cityview Drive
Rexdale, Ontario, Canada M9W 5A7

SUMMARY

This paper discusses the characteristics of a system that can determine the flight profile of an aircraft in three orthogonal co-ordinates to an accuracy of a few feet. A standard commercial quality Inertial Navigation System provides all the required aircraft dynamic and attitude data while a specially developed infra-red sensor provides periodic updates. A ruggedised digital computer efficiently implements an 18 state Kalman filter for estimation of the inertial errors. Filter data is stored on magnetic tape for immediate reprocessing by a fixed interval Bryson-Frazier smoothing algorithm that further refines the system performance. These advanced analytical techniques, applied in real time, are controlled in a multi-task environment by a powerful software operating system.

The infra-red position sensor employs two LED illuminators and two scanning linear photo-diode arrays, which, by means of retro-reflectors mounted on the ground in surveyed locations, allows the aircraft position to be determined at the point of overflight to better than a foot in all three axes.

Six months of flight trials have demonstrated the robustness and reliability of the system concept. By comparing the system performance with an independent measurement of the aircraft flight profile as determined by a photo-theodolite range, accurate assessments of the airborne performance have been made. These are presented and show that the aircraft profile can be measured to an accuracy of several feet in all three axes over a short period of time.

Applications for such a measurement capability are discussed with emphasis on the initial purpose of providing an accurate self-contained trajectory measuring system for I.L.S. and M.L.S. flight checking. Other uses such as airborne surveying and weapon-release determining systems are mentioned.

1. Introduction

There exists a large variety of applications where the instantaneous and continuous knowledge of the position of an aircraft is required to a high accuracy, usually over a limited portion of the flight profile. For normal navigational purposes for both commercial and military aircraft, accurate knowledge of position is only required during the landing phase with this function being commonly met with the use of Instrument Landing Systems, (ILS), Microwave Landing Systems (MLS) and Precision Approach Radars (PAR). The initial commissioning and subsequent calibration of these landing aids requires the use of an even more accurate system in order to reliably determine any error patterns in these landing aids. During flight certification of new or modified aircraft designs, it is necessary to accurately determine the position of the aircraft over a limited portion of the flight profile to determine take-off and landing performances. The assessment of weapon release performance and bombing accuracy similarly requires accurate knowledge of the actual flight profile of the carrier aircraft over a short distance. Other applications with similar requirements include airborne geophysical studies, airborne surveying, cargo dropping, photogrammetric operations and precision crop spraying.

In order to meet the requirements for these numerous applications, an equally numerous variety of measuring systems have been developed with characteristics usually optimised for the individual applications. Until recently, the measuring technique most frequently employed was based upon the tracking of the aircraft using optical devices, such as manually operated theodolites, infra-red tracking theodolites and photo or cine-theodolites. Such systems are line-of-sight, require good visibility, accurate setting up and calibration while their performance is dependent upon the range from the measuring device as well as the geometry in a multi-device system. Although such systems are marginally portable, they only require a minimum of airborne equipment which is an advantage for some applications. Other accurate measuring systems contain different configurations of navigation aid equipments such as multiple Distance Measurement Equipment (DME) receivers as well as updated or unupdated Inertial Navigation Systems (INS). Currently the use of laser trackers and laser rangefinders are being successfully employed in this role although they also suffer from some of the disadvantages of the optical measuring systems, namely the requirements for line-of-sight, good visibility and extensive setting up and calibration.

The subject of this paper is a system that was developed with the objectives of a higher performance and fewer restraints compared to the currently available aircraft profile measuring systems. The performance objectives were to determine the aircraft position history, regardless of any aircraft manoeuvring, out of sight if necessary from the ground, to an accuracy of a few feet in the three orthogonal axes. This performance was to be obtained using a minimum of ground equipment, no ground operating personnel, using no ground or post flight processing and to be achievable in any flyable weather down to Category II minimums. Development of such a system has been successfully completed with proven performance and the remainder of the paper discusses this system and its development and mentions its potential uses. Its initial application is for ILS commissioning and recalibration up to Categories II and III, for which purpose the development of the Inertial Referenced Flight Inspection System (IRFIS) was funded by the Canadian Ministry of Transport and implemented by Litton Systems Canada Limited.

2. System Concept

The IRFIS utilises aircraft velocity data as measured by an Inertial Navigation System (INS) to determine the instantaneous aircraft position relative to an established ground co-ordinate frame. However, the required performance can only be obtained with a velocity accuracy some fifty times better than that achievable with a commercial INS. For this reason, the IRFIS incorporates a powerful mathematical error estimator, or sub-optimal Kalman filter, in conjunction with a unique update sensor specially developed for this application. A large proportion of the inertial velocity errors are thereby estimated and removed. The residual performance, however, although much improved over the raw INS performance, would still be capable of further improvement. Consequently a second mathematical error estimator, a Bryson-Frazier smoother, is incorporated to yield a further significant performance improvement.

The system components, shown in the block diagram of Figure 1, are described below along with their respective functions as employed in the IRFIS.

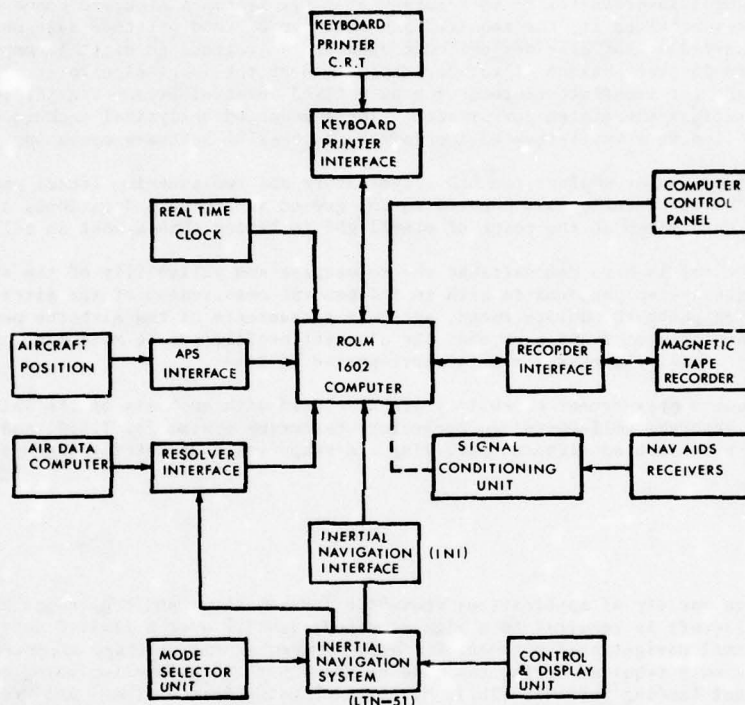


Figure 1. Inertial Referenced Flight Inspection System Block Diagram

a) Inertial Navigation System (INS)

The INS used in the IRFIS is the commercial Litton LTN-51 as used by airlines, military and scientific agencies around the world with optional dual speed attitude synchro readouts and a vertical accelerometer. Accurate navigation data is computed every 50 milliseconds and although not normally available to users at this frequency, a special buffer unit was developed to transfer the required data directly from the INS computer memory bus to the ROLM computer. Inertial velocity along three orthogonal axes is used to determine raw aircraft relative position while other INS data, direction cosines, heading etc., are used for propagating the Kalman filter error estimator. In addition, aircraft attitude, track and drift angle data from the INS are used in the update sensor mechanism to generate an accurate aircraft position at selected points during the mission profile.

The INS has the normal ARINC standard Control/Display Unit and a Mode Selector Unit such that a global navigation capability is provided by the INS as well as providing raw data for the IRFIS.

b) Aircraft Position Sensor (APS)

This update sensor, specially developed by Litton from a concept originating within the Canadian Ministry of Transport, determines the instantaneous aircraft position along three orthogonal axes to an accuracy of better than a foot. Two such accurate position measurements during each pass allow the aircraft position history to be calculated using corrected INS velocity data to interpolate between these measurements and to extrapolate beyond these points. Moreover, knowing the aircraft position accurately at two points allows both velocity and orientation errors to be positively determined by the Kalman filter.

The APS, described in more detail later on, consists of two infra-red detector arrays mounted at 90 degrees to each other and installed in the aircraft pitch plane. Associated with each detector array is an infra-red illuminator that generates a beam pattern beneath the aircraft in the same plane as the detectors. By placing pairs of retroreflectors in known locations on the ground, infra-red energy from the illuminator can be returned to the detectors as the retroreflectors are overflown. The angles formed by the aircraft to the two retroreflectors can then be calculated. When this data is considered along with the aircraft attitude, groundspeed and drift angle data from the INS, the aircraft position can be accurately computed at the instant of overflying the retroreflectors. Figure 2 shows the APS mounted below the INS.



Figure 2. LTN-51 Control Display Unit, A.P.S. Electronics Unit, and Inertial Navigation Unit with APS Camera Unit Attached

c) Computer and Software

The Rolm 1602 general purpose minicomputer is used to perform the system management and data processing functions. Floating point firmware is incorporated for fast and accurate analytical data manipulation. In the current configuration, just over 20K of the installed 32K memory is used and the computer duty cycle has adequate margin for any probable growth situations.

The software is written in Rolm Assembler language using "top-down" modular design techniques that allow efficient coding, testing and documenting of the software. The software executive organises and sequences the tasks to provide for the orderly transfer of data to and from the computer as well as ensuring the calling of the proper analytical modules at the appropriate times. The 18 state Kalman filter was designed following an extensive sensitivity analysis and simulation period and was derived from a number of Litton navigation systems employing such techniques. The Bryson-Frazier smoother, not yet widely used in such applications, takes stored Kalman filter data immediately following an update and refines the errors previously estimated by the filter in the light of the new information received at the update time. A significant performance improvement is thus obtained.

Other software analytical modules determine the aircraft update position using inertial and APS data while a conventional third order, fixed gain baro-inertial height loop is implemented to prevent an open-loop runaway of the vertical height error.

d) Operator/Computer Interface

As presently configured, the operator communicates with the system using a Keyboard for meaningful command and data entry while data output is displayed on a Visual Display Unit with a permanent record generated by a thermal printer. Operator commands are reduced to a minimum in this semi-automatic system while message outputs provide explicit statements of any detected errors or command requests. Quick-look data representing the aircraft profile for every 5 seconds of flight are printed out at the completion of each pass while the total record of aircraft position for every 100 milliseconds of flight is stored on magnetic tape for subsequent print out on the printer.

e) Magnetic Tape Recorder

Mass data storage capability is provided by a dual installation of a four track cassette magnetic tape recorder. While a calibration is being performed, blocks of Kalman filter and inertial data are transferred every 5 seconds to the tape unit. Track and tape management is performed by the software which maintains a record of spare tape available and advises when track or tape changeover is required. Data is read back from the tape for reprocessing by the smoother with final data being restored back onto the tape as well as being selectively displayed on the printer. Program and mass data loading, when required, is also effected using the cassette tape recorder.

f) Air Data Computer

An air data computer is used to provide a dual speed synchro signal representing baro-altitude to be employed by the baro-inertial height loop.

g) Interface Electronics

A Rolm expansion chassis is employed to house the system interface electronics that allow communication between the system components. A multiplexed 8 channel, 14 bit synchro to digital converter takes inertial attitude and baro altitude data while special purpose interface electronics were developed to accept inertial digital data and APS data. The standard data output rate from the INS was inadequate for this application and a buffer box was developed that allowed the INS computer data to be accessed every 50 milliseconds.

3. Method of Operation

The INS is initialised in the normal fashion prior to take-off while the system operator selects prestored data defining the reference coordinate frame in which the aircraft position will be measured. This coordinate frame is expressed as a ground zero point with an orientation defined by the location of the ground based retroreflectors. If required, en-route steering to the measurement area or range is provided by the INS.

Just prior to overflying the retroreflectors for the first time, the operator instructs the IRFIS to prepare for initialisation. The APS illuminator switches on and a measurement is received as soon as the retroreflectors are overflown. This enables the IRFIS to accurately construct the coordinate frame datum in which it will base its subsequent position measurements. A second APS measurement at the second pair of retroreflectors allows the coordinate frame orientation to be automatically determined to the required accuracy and the IRFIS is now ready for determining the aircraft relative position.

The aircraft is repositioned for a second pass with the operator defining to the system at which point in the profile the measurements should commence. Upon receiving this command, data transfer to tape commences representing the current estimate of aircraft position for every 100 milliseconds of flight between this start position and the end of the profile for which the measurements are required. The APS update sequence is now fully automatic with the aircraft flying any required profile that includes an additional overflight of the retroreflectors. Following the second update, the system automatically enters the smoothing mode, printing out the position history for the approach on the VDU and printer. Any number of repeat passes may be made, limited only by the number of spare magnetic tape cartridges carried in the aircraft. Figure 3 shows how the IRFIS is applied to the application where the aircraft profile during a landing approach is required to be accurately determined.

4. Aircraft Position Sensor (APS)

The APS/retroreflector design evolved from a study of available techniques that could permit the determination of an aircraft position to better than twelve inches in all three coordinate axes. In addition, the update sensor would have to operate in all weather, must have no hazardous characteristics, must not interfere with radio communications, must have a real time output and most important of all, must require no ground personnel or the ground installation of expensive equipment. Laser rangefinders, low range DME's and other normal position measuring devices were investigated and rejected for failure to meet one or more of the above requirements. Once the decision was made to proceed with an infra-red photodiode array as the basis of such a sensor, it was then necessary to decide upon whether a ground based or airborne source of illumination should be used. The total power and cabling costs of a ground based illuminator proved to be prohibitive and restrictive and consequently an airborne illuminator was selected.

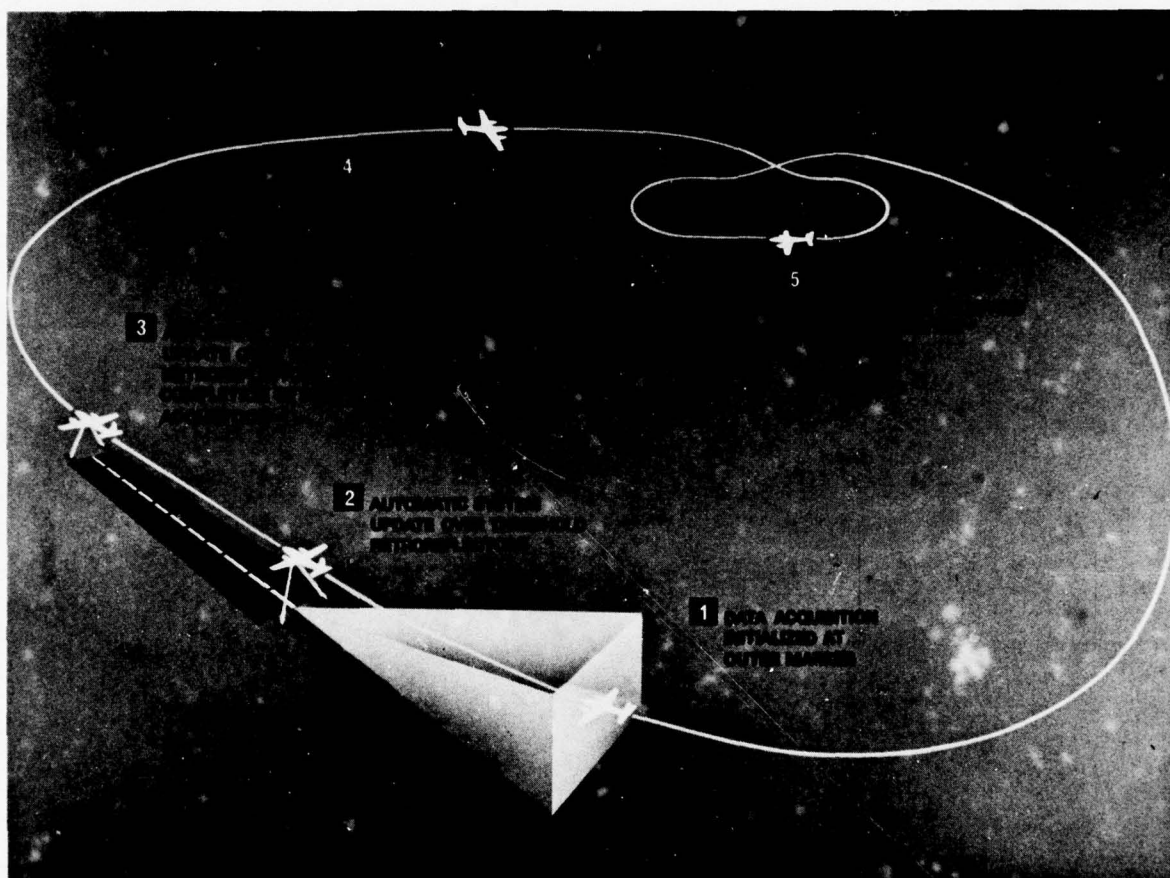


Figure 3. Typical ILS/MLS Calibration Flight

Figure 4 shows diagrammatically the final configuration of the APS while Figure 5 shows the actual equipment. Taking advantage of extensive experience and equipment used in the production of inertial quality accelerometers and gyroscopes, the APS design evolved into a solid state device with machined tolerances of tenths of thousands of an inch that allowed component interchangeability without recalibration. The 768 element linear photodiode array is cemented into a solid block of tool steel with surfaces machined parallel and orthogonal relative to the photodiode array. Two such arrays are mounted on a stable base plate and positioned in the focal plane of two high grade optical lenses.

Associated with each photodiode array is an illuminator array of 27 high power Gallium Arsenide light emitting diodes (LEDs) emitting in the non-visible spectrum at a frequency of 940 nm optimised to match the photodiode peak sensitivity. A cylindrical lens mounted in front of the LEDs generates a rectangular beam pattern approximately 45 degrees by 2 degrees such that, when installed in a aircraft flying at an altitude of 50 feet, an area 200 feet wide by 2 feet long is illuminated. Returns are received provided that the aircraft is flown within its required "window" over the retroreflectors. This window ranges from between 30 feet to 80 feet above the datum and ± 15 feet to either side of the centreline and has been demonstrated to be perfectly adequate for all normal flying conditions.

The photodiodes are sequentially interrogated at 2kHz. When one or more elements detects returned energy from the retroreflectors, its address is passed to the computer for verification. Simultaneous, or near simultaneous, detects from the pair of arrays generates diode addresses that are proportional to the angles formed by the axis normal to the aircraft longitudinal axis with each retroreflector as shown in Figure 6. This angular information, coupled with accurate aircraft attitude and dynamic data from the INS, allows the aircraft position relative to the known positions of the retroreflectors to be determined. In order to eliminate the detection of unwanted spectral reflections of sunlight off water, ice or other surfaces, the LEDs are pulsed on and off in synchronism with the photodiode interrogation frequency and all detections are subjected to a series of hardware and software criteria checks before they are validated.

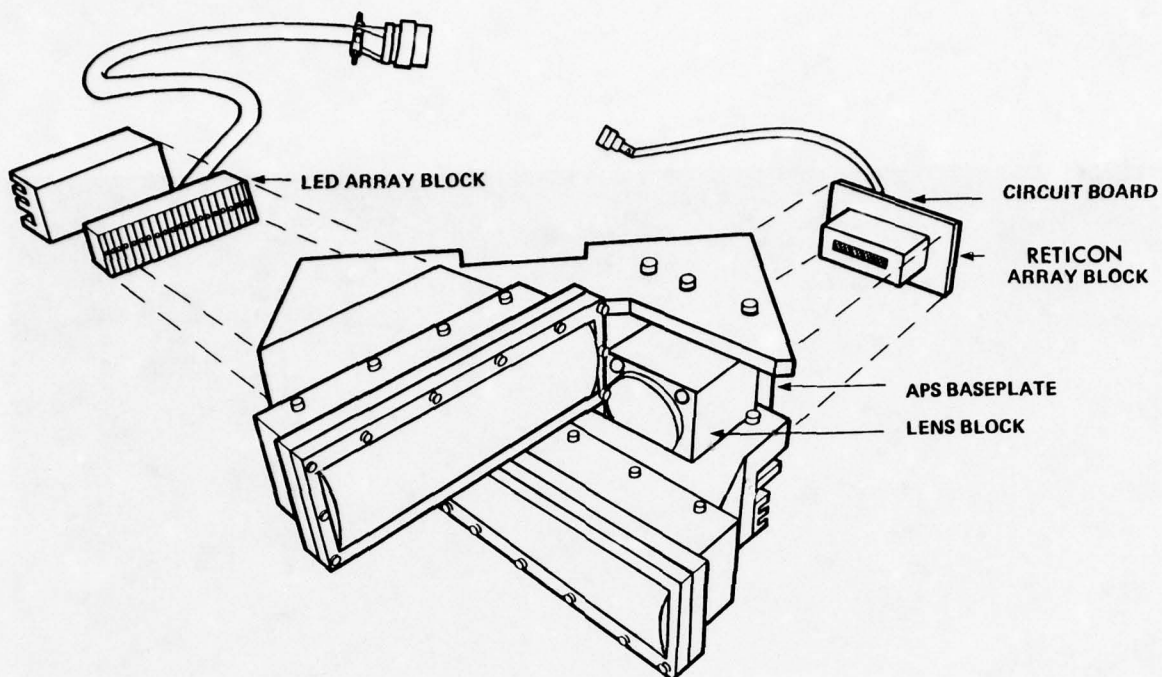


Figure 4. Diagramatic View of the Litton Aircraft Position Sensor

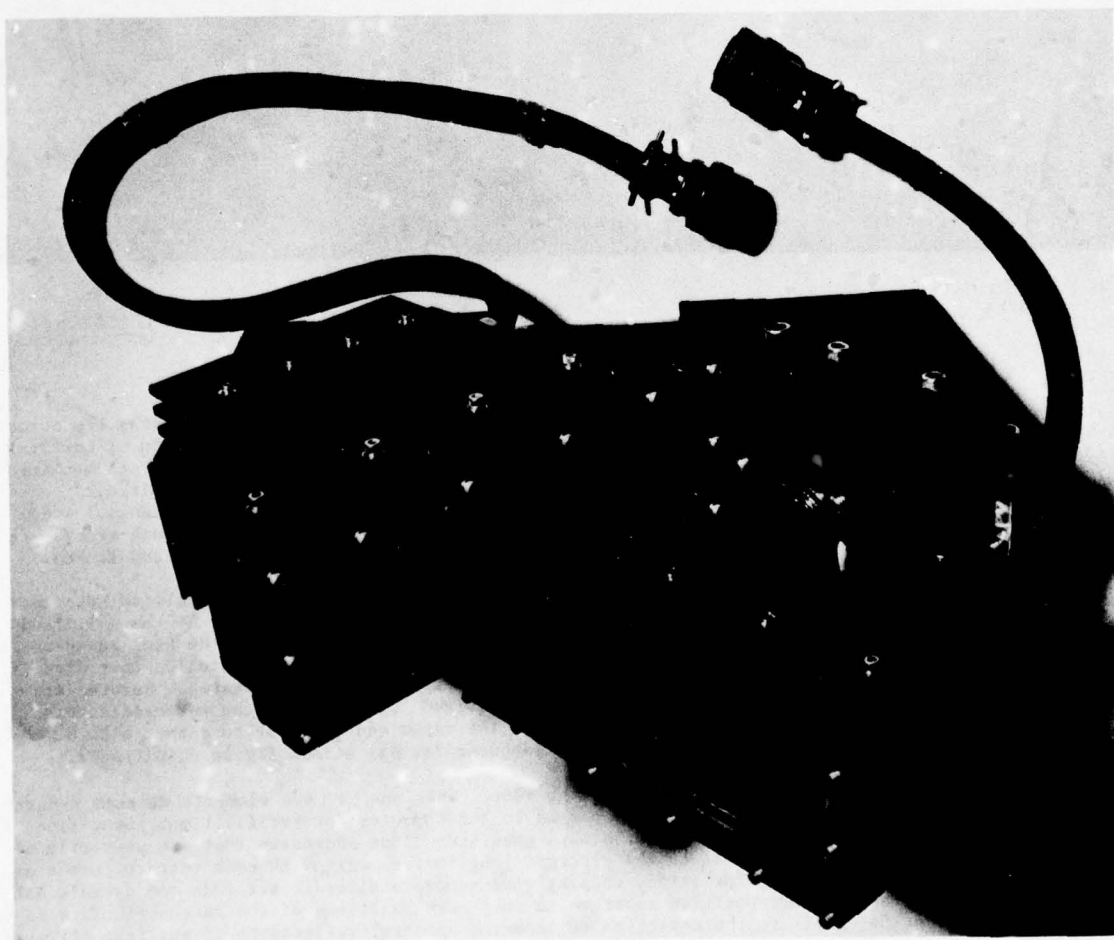


Figure 5. Aircraft Position Sensor Camera Unit

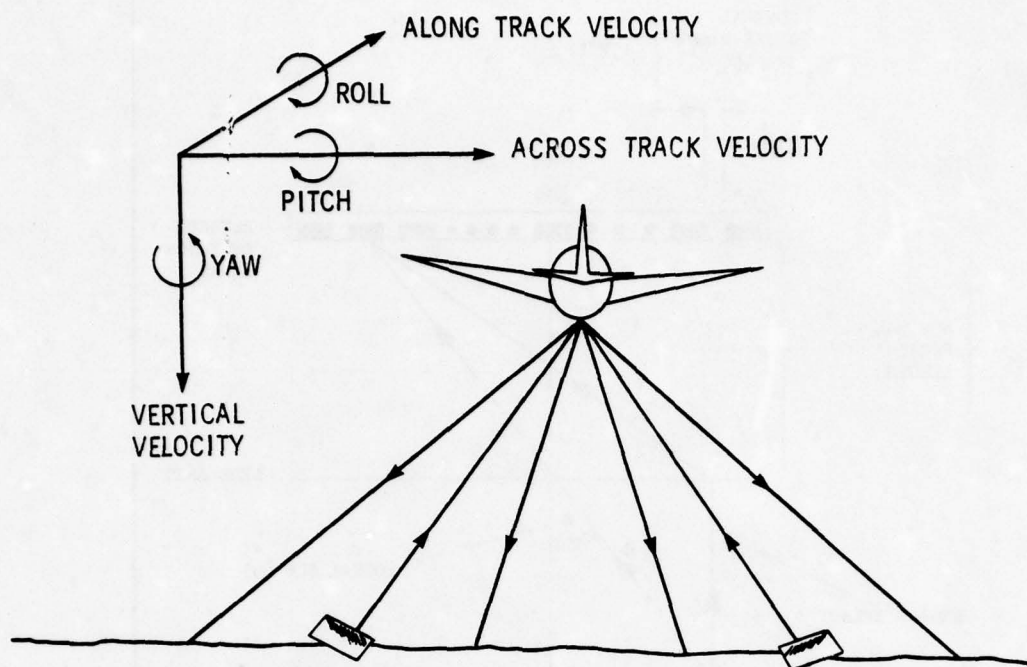


Figure 6. IRFIS Update Configuration

In order to completely calibrate the APS system, two important measurements have to be performed. Although the look-angle can be calculated simply by applying an arctangent law to the nominal focal length and energised diode displacement (Figure 7) the effect of machining tolerances, variations in inter-diode spacing and asymmetrical optics in the receiving lens would cause significant loss in the accuracy of the final aircraft Cartesian position. These problems are overcome by applying a calibration law to the ratio

$$C = F/K$$

where F = nominal focal length
 K = nominal inter-diode spacing.

This results in a curve as shown in Figure 8. The characteristics of the slope in the curve permitted a linear calibration relationship to be defined with two break-points. The calibration is performed in the laboratory using an optical cube and calibrated turntable.

The other required calibration was an accurate measurement for each detector of the nominally 45° offset to the optical axis from the vertical.

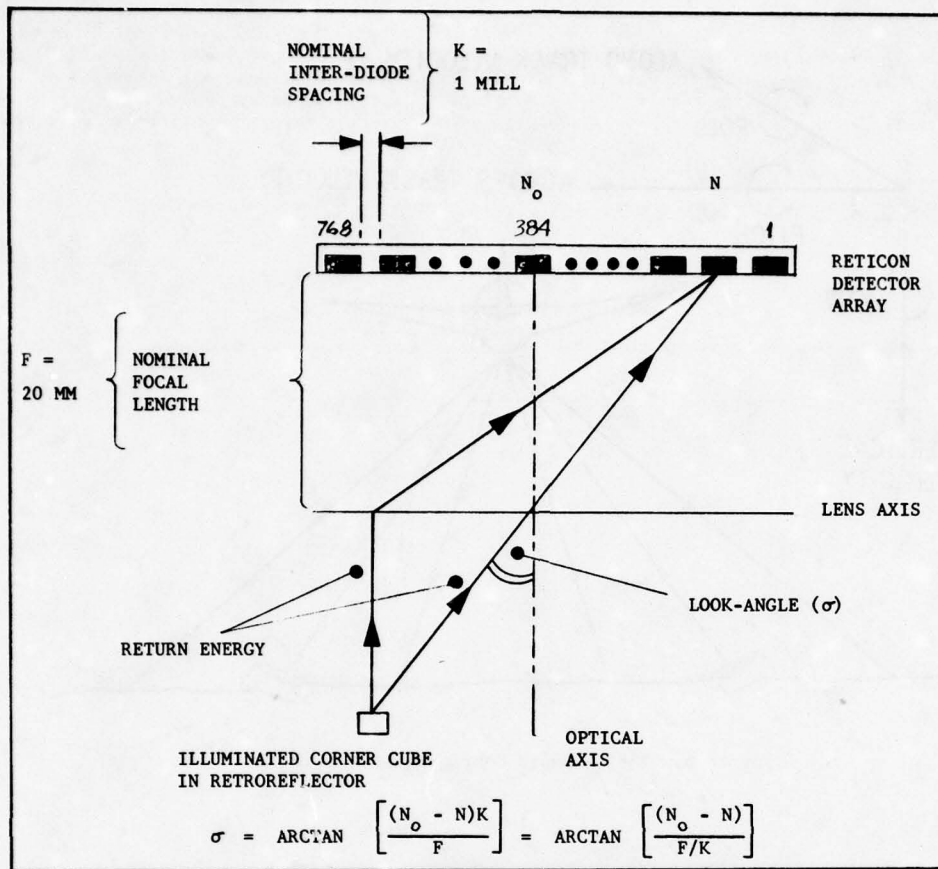


Figure 7. APS Calibration Requirement

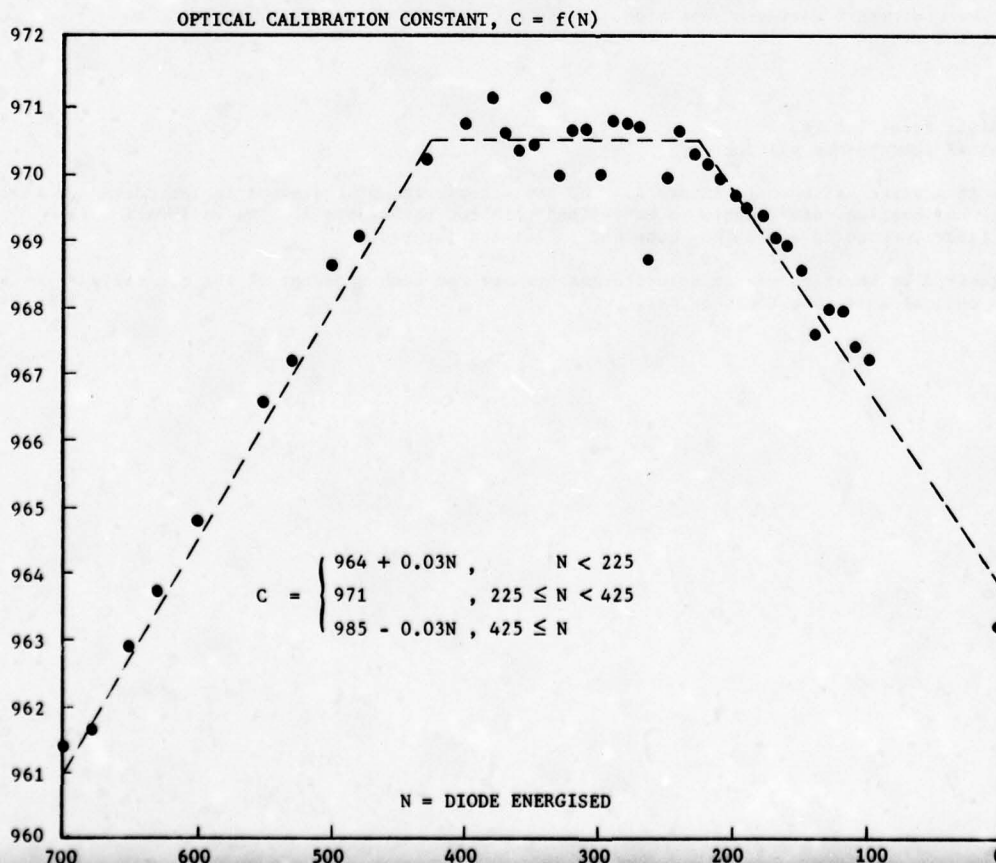


Figure 8. Optical Calibration Curve for APS Camera System

5. Kalman Filter Design

The choice of states for the Kalman filter was primarily dictated by the type of navigation aiding required to realise the varying degrees of accuracy specified for facility calibration. Secondary to this, the growth potential of the system dictated a filter design that lent itself to a relatively easy customisation. At least for Inertial Navigation filter applications, the three major areas of design interest are the Inertial Platform mechanisation errors, the Inertial Sensor mechanisation errors and the mechanisation errors of the Navigation Aids required to aid the Inertial System (if these are of a sufficiently compact and linearisable nature).

Inertial platforms today are sufficiently well understood to enable the error characteristics to be described as a set of linear differential equations. The number of equations necessary for this description depends on the choice of error states. This choice is usually dictated by the mechanical design of the platform. The complexity of the equations may be reduced by considering specific regimes of flight profile and is often limited by the computing power available. The platform states chosen for the IRFIS filter are defined in Table 1. Of particular significance is the Litton practice of using angular position error rather than geographic position error and of not including platform tilt explicitly, but rather as a state imbedded in the "PSI" equation:

$$\Psi = \phi - \delta\theta$$

By dropping the explicit tilt state, the number of necessary platform differential equations is reduced to 7.

In addition to the inertial platform states, the sensor states included in the filter are:

- (i) Level axis accelerometer residual bias after partial calibration resulting from platform leveling.
- (ii) Gyro bias residuals.
- (iii) z-axis accelerometer bias/scale-factor equivalent error

The Kalman filter was designed to be updated by position measurements in Cartesian (relative) coordinates using the APS and in Geographic coordinates for Multiple DME updates. Rather than relate the Cartesian measurement through a complex measurement matrix to the essentially Geocentric platform states modeled by the filter, it was felt expedient to augment the filter with three additional states. These states, δS , correspond to a relative grid coordinate system centered at the retroreflectors and oriented along the line joining the centres of the retroreflector pairs. Displacement errors in this coordinate frame are initialised on the first approach and there after continue to propagate in the relative grid frame. This approach has two advantages: first, it enables a more straight-forward application of the Kalman update equations and second it enables the relative position to be compensated directly by the filter estimates. The marginal increase in the complexity of the covariance equation is more than off-set by the greatly reduced computer duty-cycle loading at the time-critical update points.

The requirement of vertical position accuracy of a few feet meant that an accurate vertical loop had to be implemented. A third-order baro-inertial loop was found necessary in order to minimise any steady state height and vertical velocity errors. Although the height loop was third-order, the error characteristics (at least statistically) were adequately described by a second-order model. It was found during flight trials, however, that the vertical manoeuvres made by the aircraft caused bias shifts of several tens of feet in the baro-altimeter output. This introduced velocity errors into the height loop at precisely those points in the flight profile that highly accurate results were sought. This problem was overcome by decoupling the baro signal during those times allowing the loop to act in a free inertial mode. Whilst decoupled, precautions were taken to avoid introducing transients on loop closure. This technique reduced the velocity errors to acceptable values as was seen in subsequent flight trials.

The complete filter programmed in the airborne computer contains 18 states:-

- (i) 7 platform states ($\delta\theta_{x,y}$, $\delta v_{x,y}$, $\Psi_{x,y,z}$)
- (ii) 2 height loop states (δV_z , δh)
- (iii) 3 grid frame states ($\delta S_{x,y,z}$)
- (iv) 2 level accelerometer bias states
- (v) 1 vertical accelerometer error state
- (vi) 3 gyro bias states

Table 1. Definition of the Inertial platform error states

STATE	DEFINITION
$\delta\theta_{x,y}$	Level axis angular position error; the small angular position that the platform would have to be moved over the surface of the Earth and in directions parallel to the platform x-, y- axes for the computer to read "true".
$\delta v_{x,y}$	Level axis velocity error arising from application by the computer of incorrect gyro torquing due to an error in the knowledge of angular and Earth rate. Also includes errors due to tilt and accelerometer errors.
$\Psi_{x,y}$	Difference between platform tilt and angular position error.
Ψ_z	Summation of angular error in computed position projected into the z-axis and the deviation of the computed wander angle from its actual value.

Solution of the Riccati Equation

The 18 x 18 filter dynamics matrix was 81% sparse. This high sparsity figure prompted a programming approach that essentially programmed the filter equations explicitly rather than using a general matrix-vector manipulation routine; the objective was to minimise computation time rather than core size. The solution of the matrix Riccati equation for the filter covariance was greatly simplified by the following approach. The Riccati equation

$$\dot{P} = FP + PF^T + Q - PH^T R^{-1} HP$$

was reduced to a linear equation in P

$$\dot{P} = FP + PF^T + Q$$

by propagating only the a priori covariance P^- during the long intervals when no measurements are being taken. The a posteriori covariance was calculated at a measurement time using the relation for the optimal gain

$$K = P^- H^T [HP^- H^T + R]^{-1}$$

whence the a posteriori covariance is

$$P^+ = [I - KH]P^-$$

These are well-known relations in the field of Kalman filtering and need no explanation. It is worth noting, however, that the expression for the optimal gain involves the inversion of a square matrix of at least the same order as the measurement vector. In most applications this is only 3 x 3 and presents no great computing burden. However, in small airborne computers it is prudent to avoid such inversions if only to prevent any likelihood of ill-conditioning through truncation during the inversion process. If the measurement vector is such that, within the limitations of computational accuracy, the correlation between measurement components is small, then the inversion may be avoided altogether by considering the sequential update process of a set of scalar measurements. Such an approach was used in the IRFIS. In this case the optimal gain matrix reduces to a vector

$$\underline{K} = P^- \underline{h}^T / (\underline{h} P^- \underline{h}^T + \underline{r})$$

and the a posteriori covariance must be

$$P^+ = [I - \underline{K} \underline{h}] P^-$$

In pursuing such techniques, care must be taken to ensure that no propagation of P take place until all components of the measurement vector have been processed. Care must also be exercised in evaluating the observable difference scalars; the effect of a previous scalar update must be taken into account when processing the next component.

Ill-conditioning of the P-matrix can often occur as a result of numerical rounding effects causing the P-matrix to lose its symmetry. This problem often leads to non-convergent, or even divergent, solutions to the a priori covariance. This was overcome by propagating only the upper-half of the P-matrix thus ensuring symmetry.

To evaluate the a priori covariance several approaches were considered. The two usual methods are straight propagation of the differential equation or a transition matrix approach. The final choice was made after a careful appraisal of the dynamics of the filter in relation to the expected flight profile. It was decided that the dynamics were sufficiently linear to permit the use of a truncated transition matrix approximation with a re-evaluation of the transition matrix Φ every 5 seconds. The a priori covariance matrix may be expressed as

$$P_{K+1} = \Phi_K P_K \Phi_K^T + Q_K$$

where,

$$Q_K = \int_{t_K}^{t_{K+1}} \Phi(t_{K+1}, \tau) Q(\tau) \Phi^T(t_{K+1}, \tau) d\tau$$

Assuming that the filter dynamics matrix, F, is relatively constant over any 5 second period, the transition matrix over the same period is expressible as the power series.

$$\Phi_K = I + F_K \Delta t + \frac{1}{2!} (F_K \Delta t)^2 + \dots$$

Truncating to first order yields

$$\Phi_K \approx I + F_K \Delta t, \quad \text{where } \Delta t = 5 \text{ seconds.}$$

The a priori covariance is then expressible as

$$P_{K+1} = P_K + (F_K P_K + P_K F_K^T + Q_K / \Delta t) \Delta t + F_K P_K F_K^T \Delta t^2$$

To evaluate Q_K the 5 second interval is subdivided into 100 equal intervals of 50 milliseconds (the basic re-fresh rate of the inertial computer). Applying Euler's method to the integral, it can be shown that

$$Q_K = N Q \Delta \tau + \left(\sum_{i=1}^{N-1} i F_i Q + Q \sum_{i=1}^{N-1} i F_i^T \right) \Delta \tau^2 + O(\Delta \tau^3)$$

Since it was hypothesised that F was relatively constant over $\Delta t = 5$ seconds it is assuredly constant over $\Delta \tau = 50$ milliseconds.

$$\text{therefore } Q_K = N Q \Delta \tau + (F Q + Q F^T) \left(\sum_{i=1}^{N-1} i \right) \Delta \tau^2$$

$$= N Q \Delta \tau + \frac{N}{2} (F Q + Q F^T) (N-1) \Delta \tau^2$$

Again, to first order

$$\begin{aligned} Q_K &= N Q \Delta \tau \\ &= Q \Delta t \end{aligned}$$

So, finally, the a priori covariance matrix may be propagated according to

$$P_{K+1} = P_K + (F_K P_K + P_K F_K^T + F_K P_K F_K^T \Delta t + Q) \Delta t$$

Judicious programming techniques finally enabled the entire 18-state covariance propagation to be loaded into less than 4K (octal) core and take less than 100 milliseconds for each 5 second propagation. The calculation of the a posteriori covariance was loaded in 400 octal locations.

6. Smoothing Principle

In order to take the fullest advantage of the high accuracy of the two AFS fixes, a modified Bryson-Frazier smoothing technique was employed. The particular class of smoother used is the "fixed interval" type, that is to say, starting from a known final state, the a priori information content developed by the Kalman filter during the previous intervals is enhanced by a weighted knowledge of the a posteriori information. The weighting is in a sense inversely proportional to the forward covariance at the smoothing intervals. The fixed intervals refer to the intervals between the (historical) measurement points. It can be argued (see Figure 10) that whilst the forward covariance increases from a measurement at time t_k to one at time t_{k+1} the backward covariance possesses similar but inverse characteristics when solved in reverse time. Since the combined covariance must be contained by these two curves there must result a better overall information content. Across a measurement, the smoothed covariance must be continuous (unlike either the forward or backward covariance) since the change in information content at this point must remain the same irrespective of time if simply by the duality of cause and effect.

The modification of the Bryson-Frazier fixed interval smoothing equations was developed by Bierman and essentially enables a discrete time formulation to be made of the change in the smoother covariance across a measurement point. This is particularly useful in real-time applications since it affords a considerable reduction in computing load.

The smoothed covariance need not be computed at all. The backward operator is found from solving

$$\begin{aligned} \lambda_{K+1}(-) &= \Phi_{K+2|K+1}^T \lambda_{K+2}(-) \\ \hat{x}_{K+1}^N &= \hat{x}_{K+2}^N - P_{K+1}(-) \lambda_{K+1}(-) \end{aligned}$$

where the interval $(K+1|K+2)$ represents an interval between two successive measurements at times t_{K+1} and t_{K+2} . In the above equations λ is the backward smoothing operator and \hat{x}^N is the N-stage smoothed state estimate vector. The remaining symbols are those used in the Kalman filter namely Φ is the transition matrix over the interval, $\hat{x}(-)$ is the a priori Kalman state estimate vector, and $P(-)$ the a priori Kalman covariance matrix. Where a measurement point is encountered, the additional information is conveyed to the smoother through the equations

$$\begin{aligned} \lambda(-) &= (I - KH)^T \lambda(+) - H^T [HP(-)H^T + R]^{-1} \Phi \\ \hat{x}^N &= \hat{x}(-) - P(-) \lambda(-) = x(+) - P(+) \lambda(+) \end{aligned}$$

It is apparent that both the Kalman gain and the a priori covariance are required to evaluate λ at a measurement point. This appears to contain redundant information since the Kalman gain K was obtained from $P(-)$ during the (forward) Kalman solution. It is possible to show that the term

$$H^T[HP(-)H^T + R^{-1}] \quad \text{is expressible in terms of the gain } K \text{ as } H^T R^{-1} [I - HK].$$

However, numerically, this form of the update equation generates highly inaccurate values of λ that cause a discontinuity in \hat{x}^N . At the present time no explanation is offered for this effect other than numerical sensitivity. The equation programmed in the IRFIS is Bierman's original form and this appears to be relatively insensitive to numerical errors. Accordingly, it is necessary to store both the a priori covariance and observable difference as well as the Kalman gain for subsequent use in the smoothing equations.

It is worth noting that the only elements of λ that are affected by the update are those corresponding to the 3 gridframe states. This, together with the high sparsity of the transition matrix Φ enables a very efficient solution to be programmed:

Update Equation: 1000 octal
 Propagation: 1100 octal
 Smoother: 125 octal

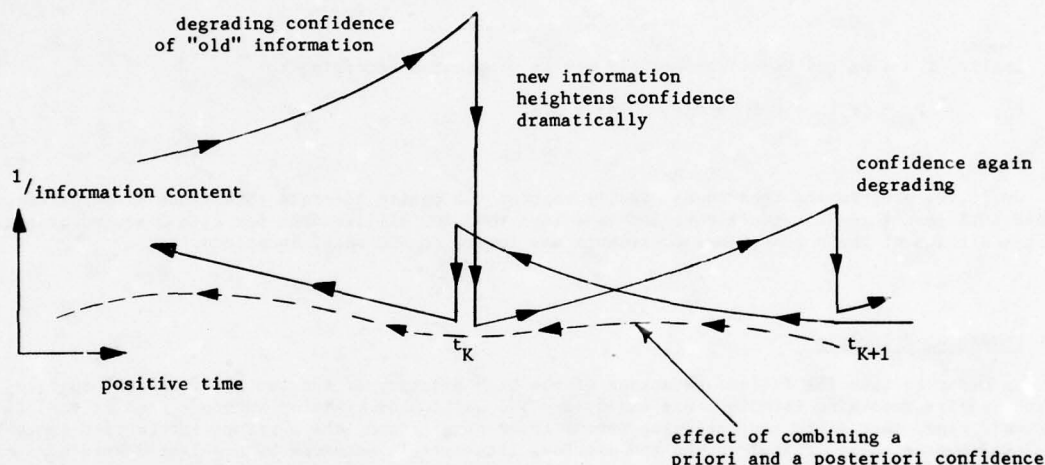


Figure 10. Intuitive Concept of Enhancement from a Smoothing Technique

7. APS Angle Reduction

The Aircraft Position Sensor system has already been described functionally. Analytically, the data received consists of a mean diode address for each of the two detector arrays together with timing, attitude and velocity information (See Figure 11). The geometry of the APS, aircraft and retroreflectors is such that a simultaneous detection can only occur with zero relative heading (i.e. zero relative track, zero drift angle). Geometrically, it can be shown that the position determination problem resolves to that of determining, for each of the two detectors, the norm of the position vector from a detector diode to a retroreflector and the three associated direction cosines. This information enables the position vector to be uniquely determined and thus the Cartesian position components can be written down by inspection.

Due to the geometry at each of the two detect instants, the direction cosines of the position vector are determined solely in terms of the angle between the nominal vertical and a detector line-of-sight. The direction cosines in platform coordinates are determined by the known transformations relating the APS set through the mounting bracket set, the aircraft body to platform gimbal set and thus to the platform axis set. The norm of each of the two vectors can be determined from a straight forward application of least squares techniques. The necessity of the least squared technique may be equivalently expressed as the statement that the sum of the squares of the direction cosines should be as close to unity as possible. A summary of the technique is shown in Figure 12.

The accuracy attainable with the APS system has been proven during flight trials to be better than 12 inches (1 sigma) in all three axis. The determining factor for this accuracy is the high quality of machining and calibration of the APS components. In order to satisfactorily establish the APS orientation relative to its reference mounting assembly, the entire subassembly underwent extensive optical calibration in the laboratory.

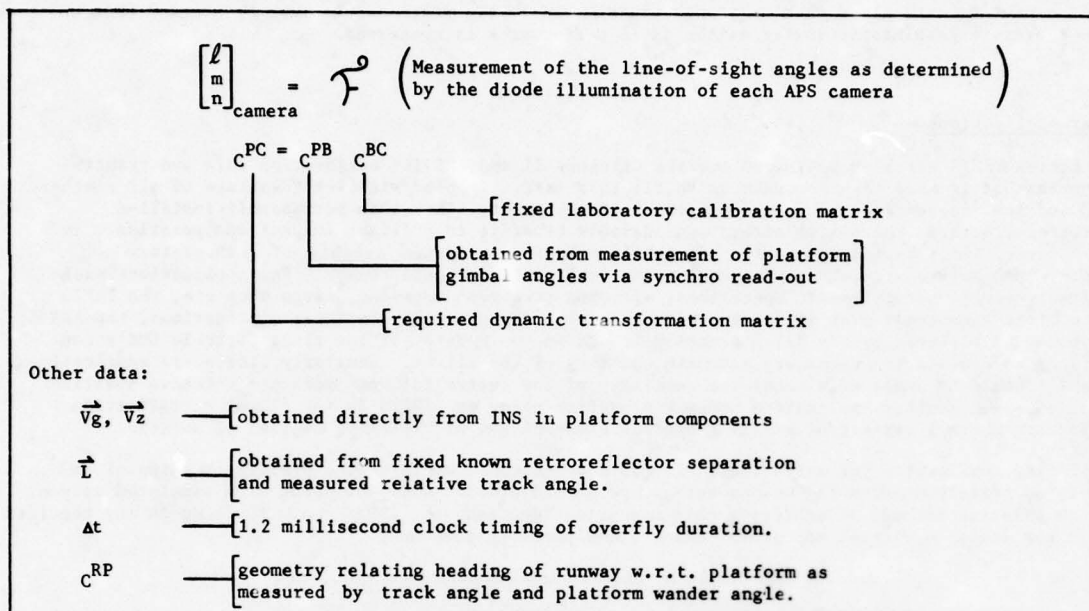


Figure 11.

The direction cosines of the position vector \vec{P} are obtained from:

$$\begin{bmatrix} l \\ m \\ n \end{bmatrix}_{\text{platform}} = C^{PC} \begin{bmatrix} l \\ m \\ n \end{bmatrix}_{\text{camera}}$$

The vector difference of the two position vectors \vec{P}_L and \vec{P}_R as measured by the left and right looking cameras during an overfly is determined from the known data on the Right Hand Side of:

$$\vec{P}_L - \vec{P}_R = (\vec{V}_g + \vec{V}_z) \Delta t + 2\vec{L}$$

Consideration of

$$\vec{P} = l|\vec{P}| \hat{i} + m|\vec{P}| \hat{j} + n|\vec{P}| \hat{k}$$

for the left and right looking cameras respectively leads to the solution of

$$\begin{bmatrix} l_L & -l_R \\ m_L & -m_R \\ n_L & -n_R \end{bmatrix} \begin{bmatrix} |\vec{P}_L| \\ |\vec{P}_R| \end{bmatrix} = \begin{bmatrix} (\vec{P}_L - \vec{P}_R) \cdot \hat{i} \\ (\vec{P}_L - \vec{P}_R) \cdot \hat{j} \\ (\vec{P}_L - \vec{P}_R) \cdot \hat{k} \end{bmatrix}$$

$$\text{viz: } \underline{A} \quad \underline{p} = \underline{d}$$

whence by least squares

$$\begin{bmatrix} |\vec{P}_L| \\ |\vec{P}_R| \end{bmatrix} = (\underline{A}^T \underline{A})^{-1} \underline{A}^T \underline{d}$$

From which solution

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\text{runway}} = C^{RP} \begin{bmatrix} l \\ m \\ n \end{bmatrix}_{\text{platform}} |\vec{P}|$$

Figure 12. Summary Technique for APS Position Fix

8. Trials Results

The system as described in this paper was subjected to extensive flight evaluation in a Ministry of Transport aircraft which explored the full potential of the system. These trials culminated with a series of flights at the photo-theodolite range at CAF Cold Lake, Alberta where an accurate assessment of the system performance was obtained. The results shown in Figures 13, 14, 15 show the results from a series of 79 flights and show the differences in aircraft position as determined by the range and the IRFIS. It cannot be determined what error was contributed by the range and it must be assumed that the plots are therefore pessimistic so far as the IRFIS performance is concerned.

9. Other Applications

The Litton IRFIS has been optimised for the Category II and III ILS calibration role and results have shown that it is more than adequate to fulfil this task. Coupled with its advantage of all weather operation and its independence of ground personnel or equipment, other than permanently installed passive retroreflectors, the system offers considerable benefits to a flight inspection operation. In essence, however, it is basically an aircraft position measuring system capable of a short-term performance commensurate with the performance of an intermittent update sensor. For applications such as aerial surveying, photogrammetric operations, airborne precision spraying, cargo drop etc, the IRFIS is able to offer advantages over existing position determining systems. In those applications, the APS would probably be replaced by the pilot eventing for an on-top update, or low range portable DME's can be integrated to provide the necessary accurate updating of the filter. Similarly, there are applications where the APS could be employed without the remainder of the system for very accurate relative position determination. Yet another application under exploration using the IRFIS is for flight certification of new aircraft where a variety of existing systems are employed with varying degrees of success.

A military application for which the IRFIS would be suitable would be the profile tracking of an aircraft to accurately measure the weapon release point and predicted impact point of a simulated weapon. Contrary to existing methods of achieving this measuring function, the IRFIS could be flown in any required profile in any location without any ground support equipment or personnel.

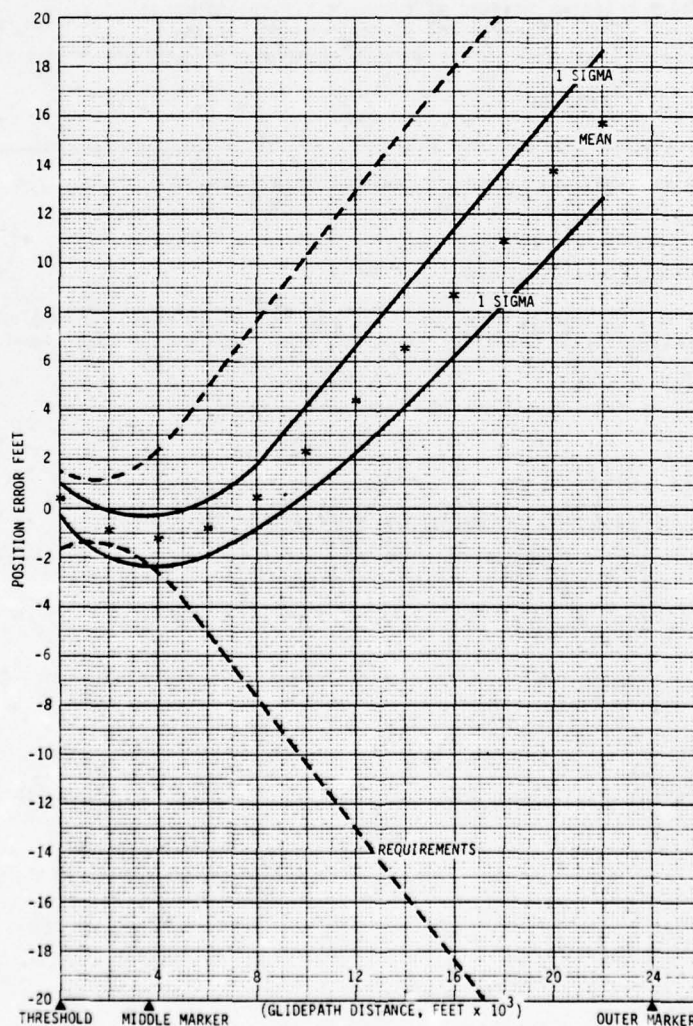


Figure 13. Across Track Position Error

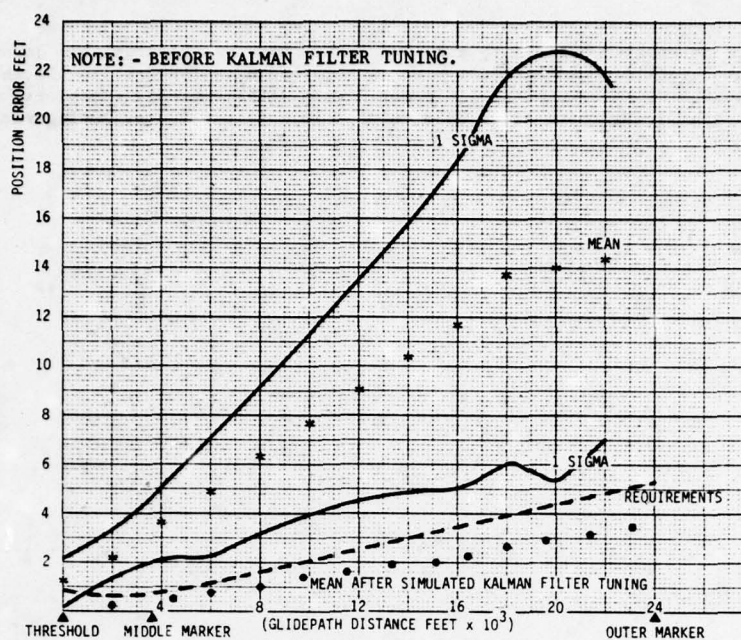


Figure 14. Above Track Position Error

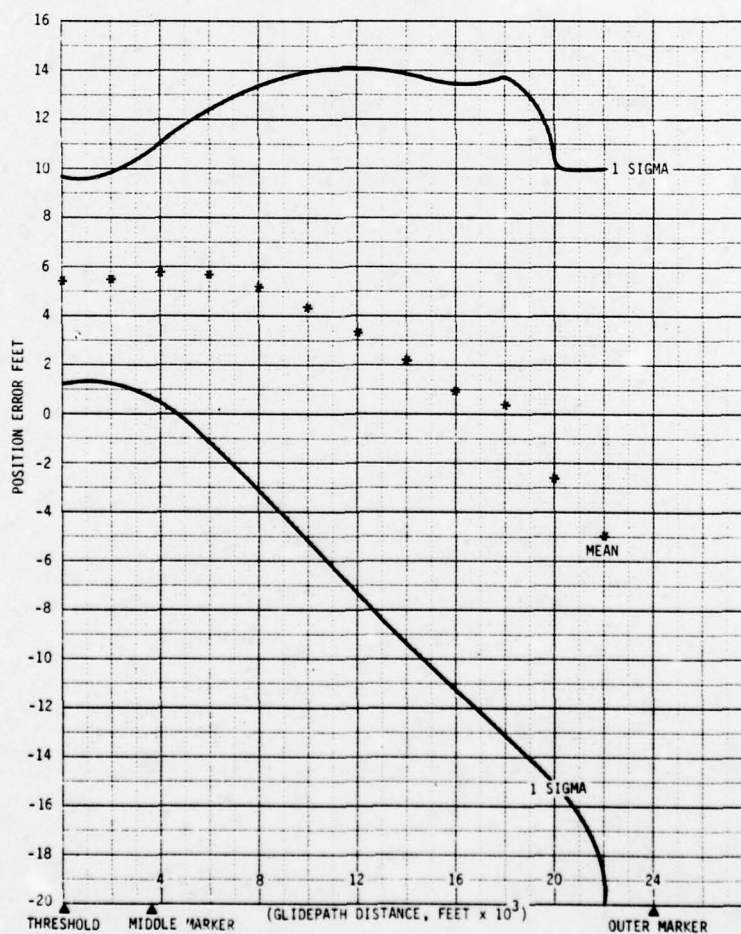


Figure 15. Along Track Position Error

10. Conclusions

The IRFIS development program has led to a unique system that has conclusively demonstrated its ability to determine the profile of an aircraft with an accuracy of a few feet in all three axes with no dependence upon ground based personnel or equipment. With different update sensors, a variety of high accuracy profile determining systems can be developed to suit particular configurations and requirements.

INTEGRATION OF FLIGHT AND FIRE CONTROL

By

Robert R. Huber, PhD
IFFC I Technical Director
Air Force Flight Dynamics Laboratory
WPAFB, Ohio
45433
USA

SUMMARY

This paper describes a US Air Force sponsored program to evaluate Integrated Flight and Fire Control (IFFC) systems in modern fighter aircraft. IFFC systems for air-to-air gunnery, air-to-ground gunnery, and bombing will be outlined. The concept involves the coupling of fire control commands into the flight control system. The goals of the IFFC system are improved accuracy, expanded employment envelope, and increased survivability. The concept will be tested on a F-15B aircraft. Primary modifications to the F-15B aircraft include the addition of a digital computer for flight control and fire control signal processing, an Electro-Optical (E-O) tracker, and a 1553A multiplex bus for communication between the F-15 Central Computer, the tracker, and the added digital computer. The paper will describe the IFFC concepts, the planned hardware implementation on the F-15B, and safety of flight considerations.

INTRODUCTION

Integrated Flight and Fire Control as defined herein is the blending of manual pilot control with automatic control for weapon delivery. Automatic control is accomplished by the coupling of the fire and flight control systems. The US Air Force Flight Dynamics Laboratory (AFFDL) and the Air Force Avionics Laboratory (AFAL) have conducted an exploratory development effort, consisting of piloted simulations at General Electric (GE), to evaluate the potential of IFFC for current Air Force fighter aircraft. Specifically, IFFC was evaluated for air-to-air gunnery, air-to-ground gunnery, and bombing. An expanded gunnery envelope, improved accuracy, increased survivability and decreased time to achieve hits are the goals of IFFC. The encouraging results of the exploratory development effort for IFFC prompted the Air Force to start an advanced development program. The effort is a joint program sponsored by the Air Force Flight Dynamics Laboratory and the Air Force Avionics Laboratory. The IFFC concept will be flight tested on an F-15 aircraft. McDonnell Douglas Aircraft is under contract to the AFFDL (IFFC I Contract) to provide the flight control system and system integration. GE is under contract to the AFAL (FIREFLY III Contract) to provide the fire control system. Consistent with the exploratory development program, the flight test will evaluate IFFC concepts for air-to-air gunnery, air-to-ground gunnery, and bombing. The objectives of the program are to: (1) demonstrate the feasibility of the IFFC concepts; (2) quantify the improvements in weapon delivery accuracy; (3) assess increase in survivability; and (4) evaluate pilot acceptance, controls and displays, stability, nonlinearities, engage/disengage transients, and failure modes and effects. The program is broken into three major tasks, which are: Task 1, Predevelopment; Task 2, Development; and Task 3, Flight Test. Task 1, which involves concept and configuration selection, was completed in April 1979. Task 2, Hardware Development, is scheduled for completion in October 1979. The flight test will start in April 1980 and continue for 15 months. The following paragraphs will describe IFFC concepts, safety of flight considerations, and F-15 modifications for IFFC.

IFFC SYSTEM OVERVIEW

The overall IFFC system block diagram is shown in Figure 1. The heart of the IFFC system is a digital computer, termed the Coupler-Interface Unit (CIU). All fire control computations are done in the CIU as well as the outer loop flight control laws. The interface between the flight control and fire control is a software interface in the CIU.

IFFC requires the development of a director fire control system. The director gunsight concept is shown in Figure 2. A director fire control system employs a sensor/tracker and ranger to measure target relative position. This information is then processed by a target state estimator (Kalman filter algorithm) to obtain an estimate of target velocity and acceleration. These estimates, along with bullet or bomb time of flight, are used to predict future target and weapon position. The director gunsight then directs a coincidence of these two positions. The error between the directed attacker weapon line position and the target line of sight is displayed on the HUD. This error is one component of the flight control coupler control laws which provide attitude rate commands to the inner loops of the analog flight control augmentation system (CAS).

TARGET STATE ESTIMATOR

A block diagram of the target state estimator is shown in Figure 3. The system was designed for a gimbaled tracker. The target state estimator is a Kalman filter with time varying gains. The estimator is formulated in a roll stabilized coordinate system. Target states estimated includes the range vector \hat{R} , range rate $\hat{\dot{R}}$, angular rate of the line of sight to the target $\hat{\omega}_t$, and target acceleration \hat{a}_t . The target acceleration is modeled as a constant angular turning rate. The roll stabilized coordinate system allows decoupling of the filter equations into a set of three filters each with three states. Kalman filter gains are computed on line for the nine state linearized system. The filter requires input covariances for range, target acceleration, and tracker noise. Computation of the director lead angle is accomplished with the filter estimates of target range, line of sight rate, and acceleration.

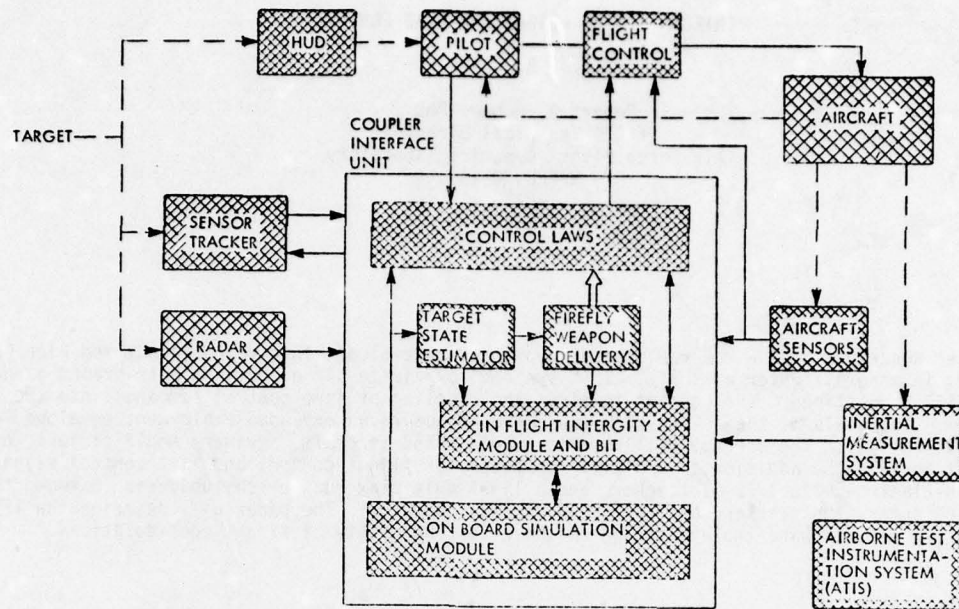


Figure 1 IFFC Functional Block Diagram ;

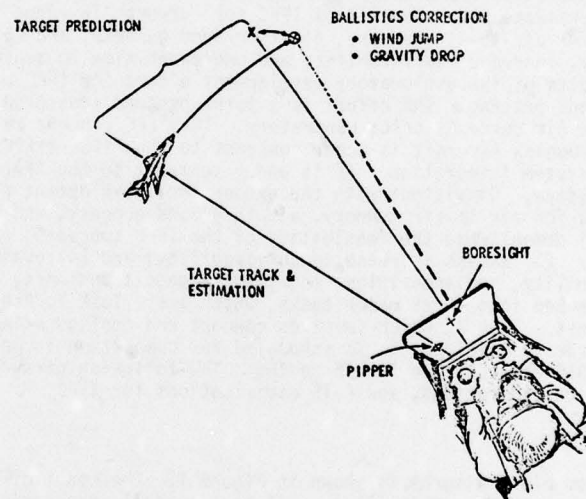


Figure 2 Director Gunsight Concept

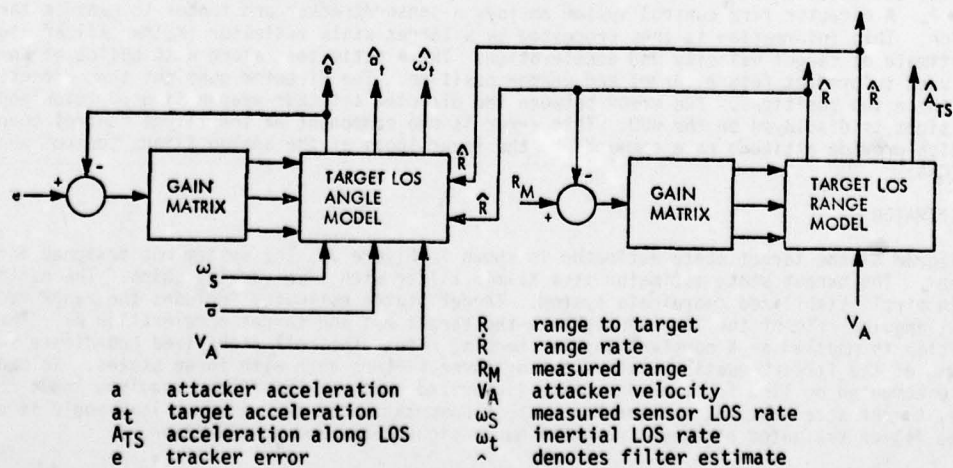


Figure 3 Target State Estimator

AIR-TO-AIR CONTROL LAWS

The technique used for automatic control in air-to-air gunnery is similar to that used in manual control. The control law rolls the aircraft so that the vector angular rate of the gun (u axis) is collinear with the predicted angular rate of the target while simultaneously nulling the traverse and elevation components of gun error. The lateral-directional control concept is shown in Figure 4.

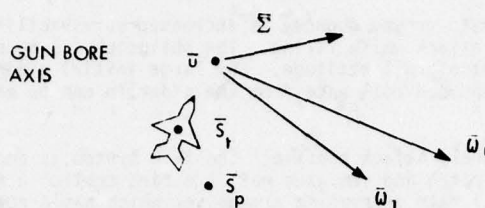


Figure 4 Lateral-Directional Control

From Figure 4, angular velocity $\bar{\omega}_1$ is the present gun angular rate vector and $\bar{\omega}_r$ is the angular rate vector of the future target position derived from the state estimator. \bar{S}_t is a unit vector along the line of sight to the target and \bar{S}_p is a unit vector along the pipper line of sight determined by the director fire control computation of lead angle. Pipper or aim error is equal to $\bar{S}_p \times \bar{S}_t$. $|\Sigma|$ is the sine of the angle between \bar{S}_p and \bar{S}_t . Roll rate command is equal to the weighted sum of $\bar{\omega}_1 \times \bar{\omega}_r$ (angle between the current gun angular rate and the future target angular rate) and $\bar{\omega}_1 \times \Sigma$ (angle between the current gun angular rate and the vector aiming error). The first term establishes a gun angular rate collinear with the angular rate of the target. The second term commands the aircraft to roll so aiming error is in the direction of turn which is nulled by changing the turning rate magnitude.

The directional control axis maintains turn coordination during the acquisition phase of the engagement. Control laws that provide high bandwidth precision pointing of the yaw axis are faded in when lateral aim error is small.

The inner loops of the F-15 CAS are modified for increased performance. The bandwidth of the roll inner loop on the F-15 is increased to support the IFFC outer loops. This is accomplished by increasing the proportional gain on roll rate feedback. In the F-15 only the differential tail is commanded by the roll inner loops. The pitch inner loops will be modified from a C* feedback to a pitch rate feedback. Precise control of pitch rate is required for high bandwidth automatic nulling of aiming error. In the yaw inner loop, washed-out yaw rate is replaced by a high gain yaw rate feedback. Loop gains are scheduled to maintain a constant bandwidth over the flight envelope.

Figure 5 is a schematic of the IFFC system for air-to-air gunnery. The terms q_r and r_r are the future pitch and yaw angular rates of the target line of sight (LOS) determined from the target state estimator. These terms provide the steady state pitch and yaw inner loop commands and are used to prevent standoff error in the pitch and yaw axes, eliminating the requirement for integral control. Gun aiming errors (Σ_v , Σ_w) plus the future LOS rate provide the commands to the pitch and yaw inner loops. The command to the roll rate inner loop is comprised of the cross product terms discussed above.

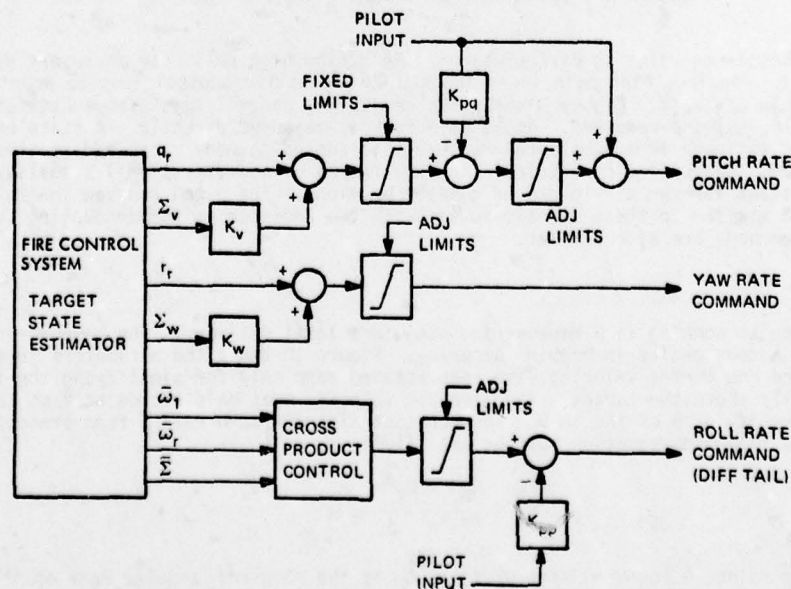


Figure 5 Air-to-Air Gunnery Control Laws

The pilot stick input in both the roll and pitch channels is subtracted from the IFFC command to cancel the normal control inputs to the horizontal tail. The stick inputs still command the mechanical system; however, the IFFC inner loops cancel these inputs except for short transients. The adjustable limits determine the authority given to the IFFC system. Pilot inputs are cancelled until the adjustable limits are exceeded. The pilot can override the system when his inputs exceed these limits. The pilot can aid or bias gun aim error by changing the gains K_{pq} and K_{pp} so that inputs are not cancelled. Pilot aiding can be useful in removing fire control errors or system biases.

AIR-TO-GROUND GUNNERY CONTROL LAWS

The emphasis for IFFC in air-to-ground gunnery is increased survivability by eliminating the traditional straight in segment of an attack while firing. The philosophy is to point the gun via commands to the pitch and yaw axes independent of roll attitude. For large initial aiming errors substantial sideslip angles can result; however, the induced roll rate from the sideslip can be effectively used to generate an evasive maneuver.

Figure 6 shows a typical gunnery attack profile. The IFFC system is engaged nearly inverted with substantial lateral aim error. The pitch and yaw axes null the fire control errors rapidly, generating sideslip. The sideslip induces a roll rate generating a maneuver which has a continually changing "g" vector direction. This type of maneuver is very effective in defeating anti-aircraft artillery (AAA) guns utilizing linear predictors in their fire control systems. Figure 7 depicts aircraft motion during such a maneuver. As shown, substantial displacement can be generated during the time of flight of AAA weapons. Typical forces encountered during an IFFC profile are shown in Figure 8. The forces are relatively constant because of the coupling between angle of attack and sideslip.

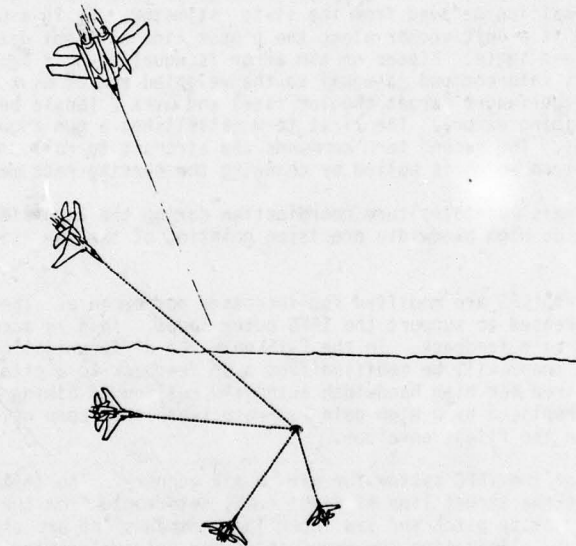


Figure 6 Typical Air-to-Ground Gunnery Attack

High bandwidth precision pointing of the weapon time during high roll rate maneuvers with large angles of attack and sideslip requires high gain inner loops plus decoupling control laws to provide a nearly neutrally stable aircraft. Figure 9 shows the decoupling control laws used. Estimates of angle of attack (α) and sideslip (β) are required. Angle of attack is measured directly. A state estimator then provides the required estimate from additional measurements including body rates and accelerations, attitude and airspeed. Decoupling coefficients are the ratios of non-dimensional stability derivatives which are nearly constant for the air-to-ground gunnery envelope. The pitch and yaw inner loops for air-to-ground gunnery are similar to those for air-to-air with the addition of the decoupling feedbacks. The pitch and yaw rate commands are also similar.

BOMBING CONTROL LAWS

The IFFC approach to bombing is a maneuvering non-wings level delivery. The maneuvering delivery is accomplished without a degradation in bombing accuracy. Figure 10 shows the parameters involved in the bombing concept. Wind and target velocity have been assumed zero only for simplifying the illustration. \bar{P} is a point vertically above the target. The aircraft velocity must be directed at \bar{P} at the time of release. \bar{G} is the gravity drop of the bomb. For constant aircraft turn rate $\bar{\omega}$ from present position to the bomb release, the following equation must be satisfied:

$$\bar{S}_v \times \bar{S}_p = \frac{R_p^2 - R_R^2}{2V_a R_p} \bar{\omega}$$

If the approach is to select \bar{G} (or bomb time of fall) $\bar{\omega}$ is the required angular rate of the aircraft to arrive at the selected release condition.

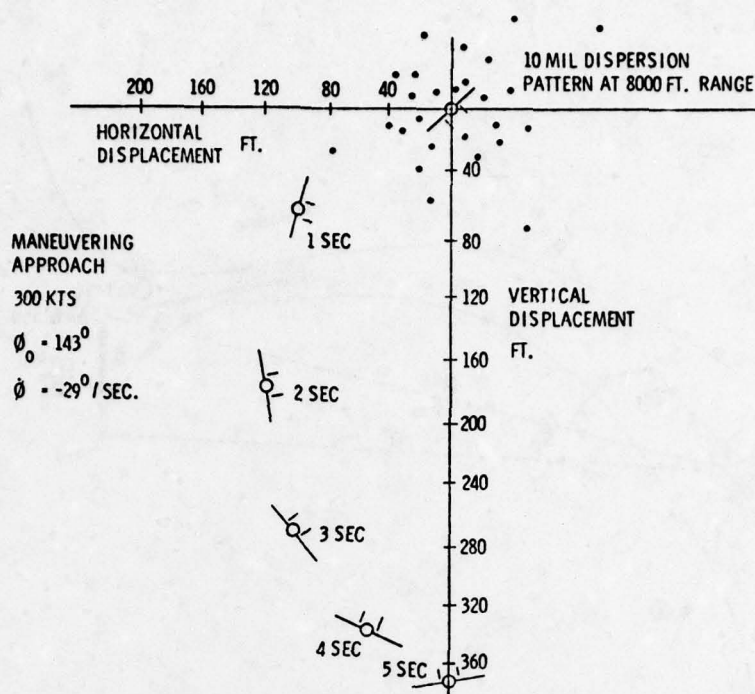


Figure 7 Displacement During Gunnery Profile

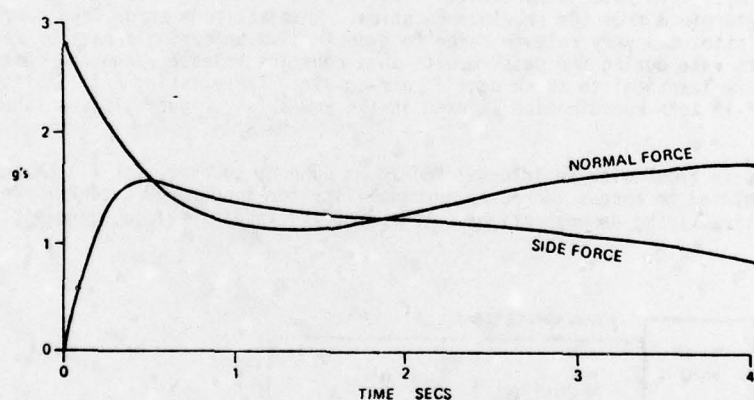


Figure 8 Force Levels in a Rolling Maneuver

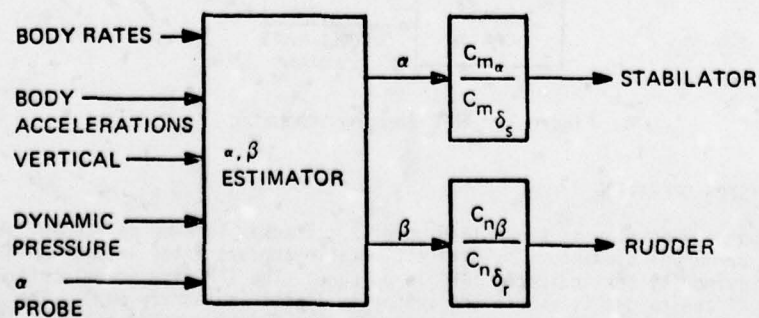


Figure 9 Neutral Stability Control

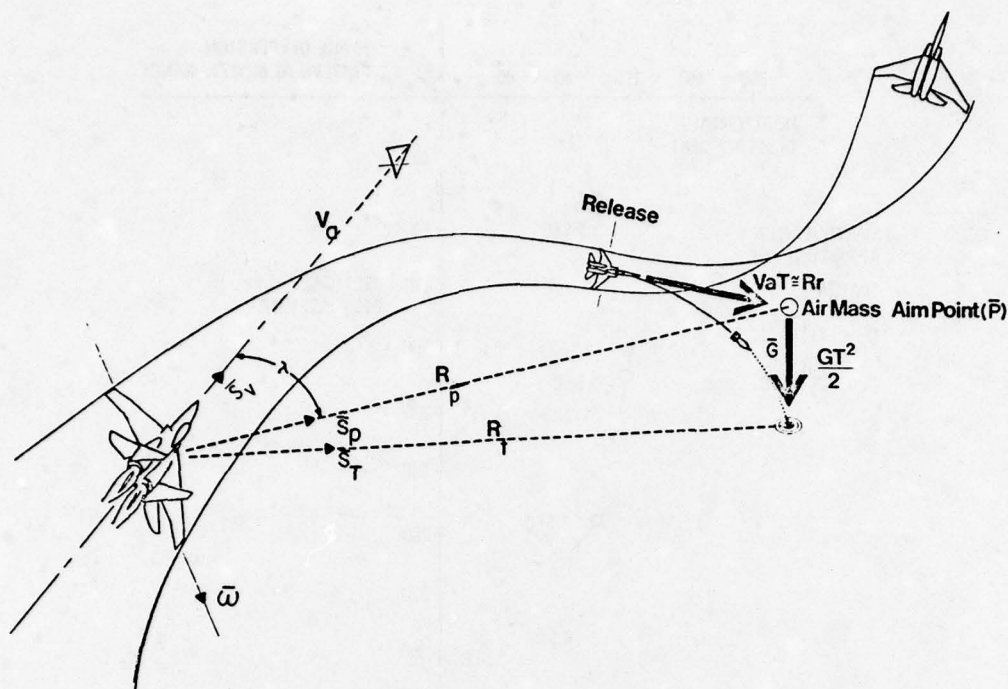


Figure 10 Bombing Relationships

The IFFC bombing concept does not preselect \bar{G} . It determines release conditions as a function of turn rate magnitude, $|\omega|$. Turn rate is controlled via pilot "g" command. The direction of the turn rate or roll attitude is determined from the previous equation. Roll attitude error then commands the roll rate inner loop. The pilot can vary release range to a desired value during a pass by varying his "g" command. Constant turn rate during the pass results in a constant release range and constant bank angle. The roll inner loops are identical to those used in air-to-air. The existing F-15 CAS is used for pitch control and existing F-15 turn coordination is used in the yaw axis. Figure 11 is a schematic of the IFFC bombing system.

The F-15 aircraft is armed with an internal M61 20 mm cannon; however, for flight test the GAU-8 30 mm cannon will be simulated to assess increased survivability for long range attacks. The M61 will be used as closer ranges for live firing demonstrations. The payoff for additional gun range is increased aircraft survivability.

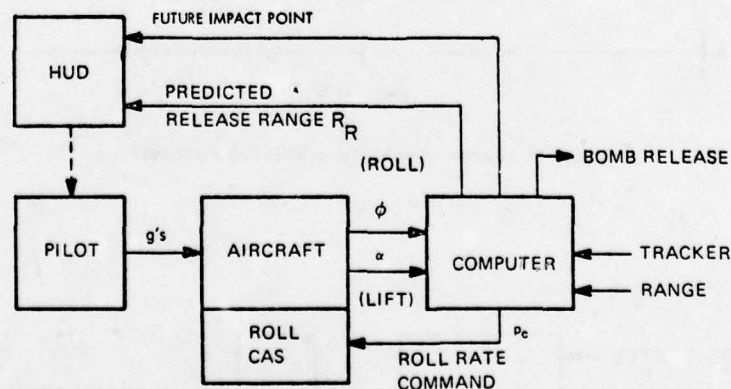


Figure 11 IFFC Bombing Schematic

AIR-TO-AIR GUNNERY SYSTEM OPERATION

The air-to-air gunnery display is shown in Figure 12. Tracker lock-on is accomplished by slaving the tracker to the radar or manual slewing. The director reticle appears after lock-on. The IFFC system can now be engaged. The authority box indicates IFFC is engaged. The IFFC system automatically holds the pipper on the target if the target is within the authority limits (authority box). The authority box defines the authority limit of the IFFC system commands to the flight control system. IFFC system authority used is displayed as the distance between the pipper and the center of the authority box. The pilot can

reduce IFFC system commands if desired by matching gross IFFC commands, especially in pitch. The pilot does this by applying the stick commands necessary to center the authority box. Using this technique allows the system to operate with reduced authority if desired.

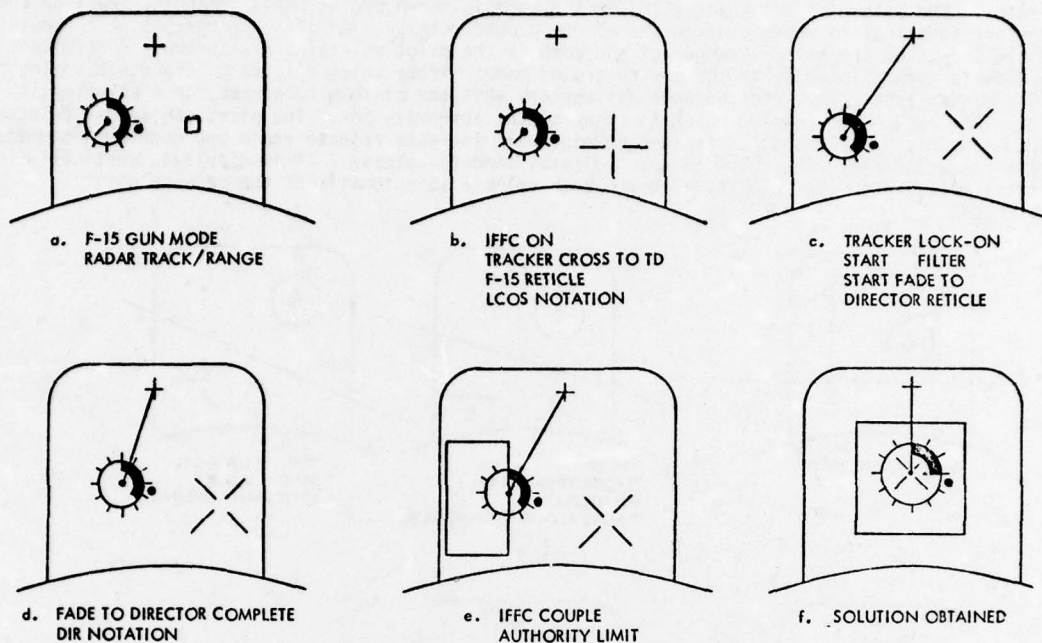


Figure 12 IFFC Air-to-Air Gunnery Displays

AIR-TO-GROUND GUNNERY SYSTEM OPERATION

Figure 13 shows the air-to-ground gunnery display. A velocity vector symbol and pull-up cue are added symbols. The pull-up cue is referenced to the velocity vector. The pull-up cue computations calculate a safe recovery altitude assuming a system failure occurs at the present aircraft attitude. To obtain tracker lock-on, the pilot either slews the tracker manually or is slewed automatically to INS coordinates. After lock-on, the director reticle appears and the IFFC system can be engaged. The authority box is shown when IFFC is engaged. The IFFC system then rapidly nulls aim error. The sideslip generates induced roll rate. The pilot can modify this roll rate to change the shape of the maneuver.

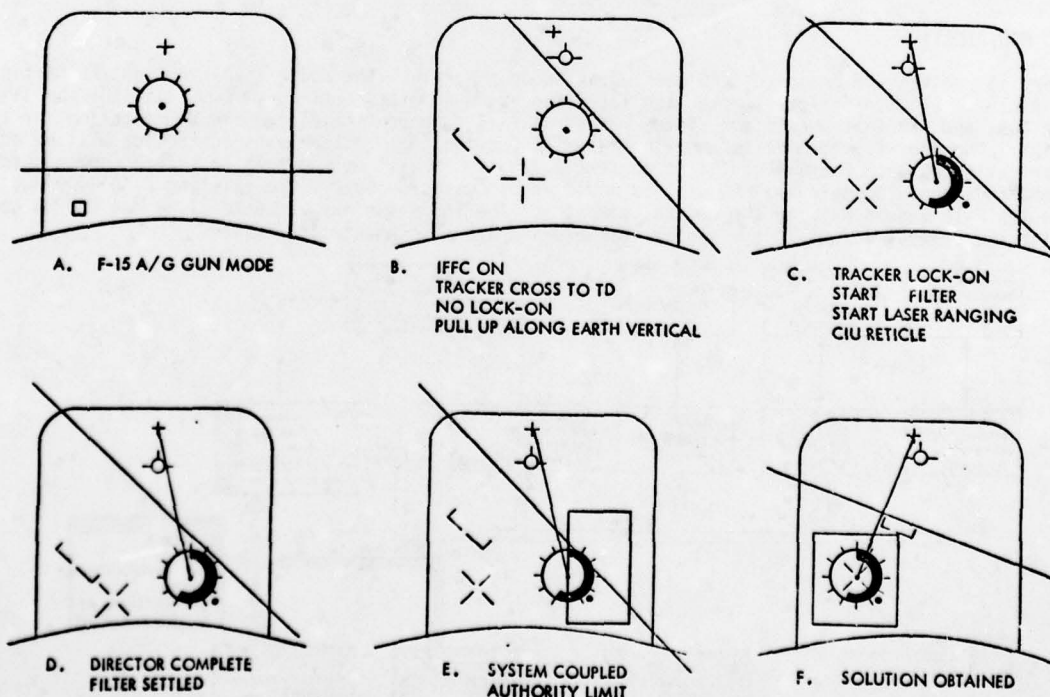


Figure 13 Air-to-Ground Gunnery Displays

BOMBING SYSTEM OPERATION

Figure 14 shows the IFFC bombing display. The reticle with the analog range bar is centered around the velocity vector. The future impact point (FIP) of the bomb is shown by the small triangle. Pull-up cue computations are identical to those used in the air-to-ground gunnery. A release range dot on the outside of the reticle indicates the release range for the bomb if the pilot maintains his current "g" command. Tracker lock-on is accomplished as in the air-to-ground mode. After tracker lock-on, the range analog bar and the FIP are displayed. The release range dot appears when the minimum turn rate for a solution is established. IFFC can now be coupled which is shown by the authority box. The pilot can adjust release range by adjusting his "g" command. Increased command will increase release range and decreased command will decrease release range. The solution cue indicates time-to-release. It is displayed vertically in HUD coordinates relative to the velocity vector. Bomb release is automatic at the release point.

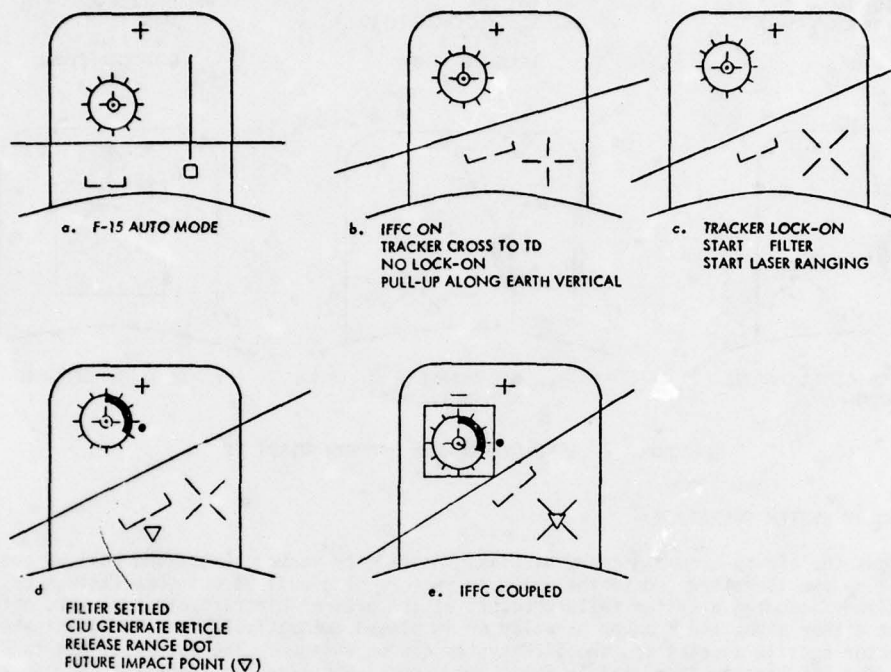


Figure 14 Bombing Displays

F-15 IFFC MODIFICATIONS

Figure 15 depicts the proposed IFFC mechanization on the F-15. The added items include the Martin Marietta ATLIS II electro-optical tracker and laser ranger, the digital coupler interface unit (CIU), the 1553A multiplex bus, and the instrumentation system. Several F-15 components will require modification for the IFFC system. The HUD will be modified for increased IFFC symbology. A 1553A bus controller will be added to the central computer. The analog flight control system will be modified to accept commands from the CIU and the inner loops in the CAS will be modified as outlined above. The existing F-15 H009 bus transmits the F-15 sensor data to the central computer. The 1553A bus was added to allow the CIU to communicate with the central computer, the sensor tracker, and the instrumentation system.

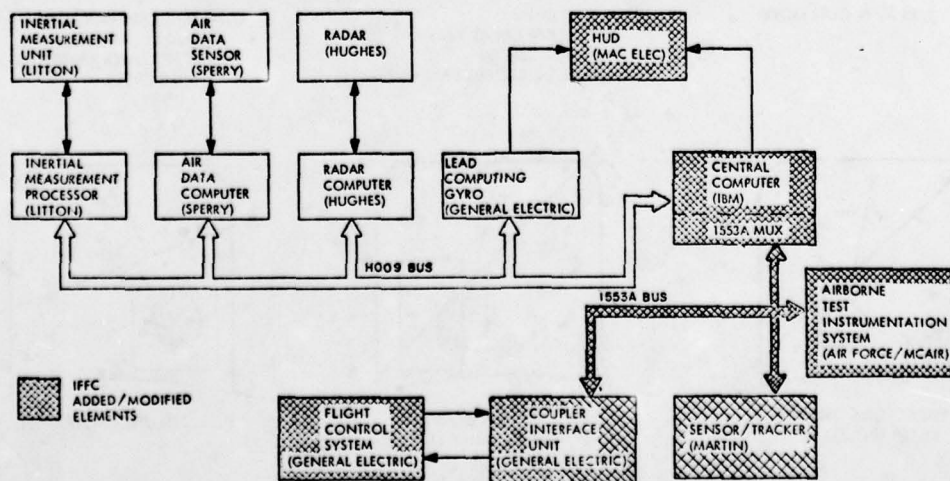


Figure 15 IFFC Hardware Schematic

IFFC SAFETY MECHANIZATION

Safety of flight is a primary consideration in the IFFC system design. A safety program will: (1) identify potential hazards within the IFFC system; (2) assess the risk associated with each hazard; (3) list the means by which each hazard can be recognized; and (4) describe the corrective action which can be taken for each identified hazard. Key safety features of IFFC are:

- (a) Provide two hands-on techniques for system disengage.
- (b) Provide authority limiters both in the CAS and the CIU to control the magnitude and rate of change of IFFC commands to the flight control system.
- (c) Provide inflight integrity monitoring (IFIM) for automatic fault detection and system disengage.
- (d) Display a break "X" on the HUD and disengage IFFC at a range which will allow a safe recovery taking into consideration the attitude of the aircraft and the worst case hardover input prior to disengagement.

The safety design features are implemented as both hardware and software. Details of the safety mechanization are discussed in the following paragraphs.

The Identification Friend or Foe (IFF) interrogate switch on the right throttle is used for IFFC couple. The couple switch is enabled only if: (1) the internal self check logic of the CIU is satisfied; (2) the F-15 central computer monitor indicates that the CIU is functioning properly; and (3) the pilot has placed the magnetically held IFFC select switch to the IFFC mode. The couple switch sends signals to the CIU and modified CAS. The CIU then sends the IFFC commands into the CAS. The CAS accepts the CIU commands by closing solid state switches. After decouple, control is returned to the standard F-15 CAS. The stick mounted paddle switch provides a second hands-on decouple technique.

The inflight integrity monitor (IFIM) will perform tests to detect failures in IFFC hardware. A summary of the functions to be performed by IFIM follows:

- (a) Monitor power supplies, internal circuitry, display generation, and input/output capability of the CIU for failures;
- (b) Identify loss or out-of-limit condition of any critical input parameter necessary to accurately calculate an attack mode solution;
- (c) Monitor control law switching and flight control commands from the CIU to insure that no signals are generated when the couple enable signal is not present;
- (d) Compare command signals issued by the CIU to command signals received by the CAS to insure that they are equal;
- (e) Compare the operating attack mode of the CIU and CAS to insure that both are using the same mode;
- (f) Monitor critical points within the CAS control law switching hardware to insure that the CAS has correctly switched to the IFFC mode and is ready to receive CIU flight control inputs;
- (g) Monitor laser output to insure that the laser is on only in the correct attack mode and only when the central computer verifies that cockpit switch positions are consistent for laser operation; and
- (h) Monitor E/O tracker inputs during air-to-ground deliveries to inhibit laser firing if tracker break lock occurs.

If IFIM detects a failure, the CIU flight control commands to the CAS will be disconnected and the CAS returned to the F-15 mode.

ACKNOWLEDGEMENTS

The majority of the information for this paper was extracted from Technical Reports and presentation material, References (1), (2), and (3), generated under the following Air Force contracts: AFFDL FIREFLY II Contract with General Electric; AFFDL IFFC I Contract with McDonnell Aircraft; and AFAL FIREFLY III Contract with General Electric.

REFERENCES

- (1) R.A. Bell, et al, Integrated Flight/Fire Control System (FIREFLY II), Vol I, System Definition, General Electric Company, AFFDL-TR-78-172, Dec 78.
- (2) Preliminary Design Review, IFFC Fire Control System, FIREFLY III, AFAL Contract F33615-78-C-1489, presentation material, Feb 6, 1979.
- (3) R.J. Landy, Integrated Flight/Fire Control Phase I, Interim Technical Report, AFFDL Contract F33615-78-C-3601, AFFDL-TR to be published.

REPORT DOCUMENTATION PAGE			
1. Recipient's Reference	2. Originator's Reference AGARD-CP-272	3. Further Reference ISBN 92-835-0247-7	4. Security Classification of Document UNCLASSIFIED
5. Originator	Advisory Group for Aerospace Research and Development North Atlantic Treaty Organization 7 rue Ancelle, 92200 Neuilly sur Seine, France		
6. Title	ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQUES		
7. Presented at	the Guidance and Control Panel Symposium held in Ottawa, Canada, 8-11 May 1979.		
8. Author(s)/Editor(s) Various	9. Date August 1979		
10. Author's/Editor's Address Various	11. Pages 372		
12. Distribution Statement	This document is distributed in accordance with AGARD policies and regulations, which are outlined on the Outside Back Covers of all AGARD publications.		
13. Keywords/Descriptors			
Digital techniques High integrity Failure detection		Strap-down Digital flight controls	
14. Abstract			
<p>The Proceedings include papers presented at a symposium of the AGARD Guidance and Control Panel, held in Ottawa, Canada, 8-11 May 1979.</p> <p>Thirty papers were presented on the following topics:</p> <ul style="list-style-type: none"> • Functional design concepts, trends, and requirements. • Advances in analytical and design techniques. • Advances in digital system design and architecture to assure high integrity. • Data processing and computation techniques. • Software design validation techniques including simulation. • Operational and system development experience. 			

<p>AGARD Conference Proceedings No.272 Advisory Group for Aerospace Research and Development, NATO ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQUES Published August 1979 372 pages</p> <p>The Proceedings include papers presented at a symposium of the AGARD Guidance and Control Panel, held in Ottawa, Canada, 8-11 May 1979.</p> <p>Thirty papers were presented on the following topics: Functional design concepts, trends, and requirements; Advances in analytical and design techniques; Advances in digital system design and architecture to assure high</p> <p>P.T.O.</p>	<p>AGARD-CP-272</p> <p>Digital techniques High integrity Failure detection Strap-down Digital flight controls</p>	<p>AGARD Conference Proceedings No.272 Advisory Group for Aerospace Research and Development, NATO ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQUES Published August 1979 372 pages</p> <p>The Proceedings include papers presented at a symposium of the AGARD Guidance and Control Panel, held in Ottawa, Canada, 8-11 May 1979.</p> <p>Thirty papers were presented on the following topics: Functional design concepts, trends, and requirements; Advances in analytical and design techniques; Advances in digital system design and architecture to assure high</p> <p>P.T.O.</p>	<p>AGARD-CP-272</p> <p>Digital techniques High integrity Failure detection Strap-down Digital flight controls</p>
<p>AGARD Conference Proceedings No.272 Advisory Group for Aerospace Research and Development, NATO ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQUES Published August 1979 372 pages</p> <p>The Proceedings include papers presented at a symposium of the AGARD Guidance and Control Panel, held in Ottawa, Canada, 8-11 May 1979.</p> <p>Thirty papers were presented on the following topics: Functional design concepts, trends, and requirements; Advances in analytical and design techniques; Advances in digital system design and architecture to assure high</p> <p>P.T.O.</p>	<p>AGARD-CP-272</p> <p>Digital techniques High integrity Failure detection Strap-down Digital flight controls</p>	<p>AGARD Conference Proceedings No.272 Advisory Group for Aerospace Research and Development, NATO ADVANCES IN GUIDANCE AND CONTROL SYSTEMS USING DIGITAL TECHNIQUES Published August 1979 372 pages</p> <p>The Proceedings include papers presented at a symposium of the AGARD Guidance and Control Panel, held in Ottawa, Canada, 8-11 May 1979.</p> <p>Thirty papers were presented on the following topics: Functional design concepts, trends, and requirements; Advances in analytical and design techniques; Advances in digital system design and architecture to assure high</p> <p>P.T.O.</p>	<p>AGARD-CP-272</p> <p>Digital techniques High integrity Failure detection Strap-down Digital flight controls</p>

<p>integrity; Data processing and computation techniques; Software design validation techniques including simulation; Operational and system development experience.</p> <p>ISBN 92-835-0247-7</p>	<p>integrity; Data processing and computation techniques; Software design validation techniques including simulation; Operational and system development experience.</p> <p>ISBN 92-835-0247-7</p>
<p>integrity; Data processing and computation techniques; Software design validation techniques including simulation; Operational and system development experience.</p> <p>ISBN 92-835-0247-7</p>	<p>integrity; Data processing and computation techniques; Software design validation techniques including simulation; Operational and system development experience.</p> <p>ISBN 92-835-0247-7</p>